EESTI MAAÜLIKOOL
Tartu Tehnikakolledž

**Kaljo Valk**

# Automaatne kuiva kassitoidu jaotur

*Automatic dry cat food dispenser*

Rakenduskõrghariduse lõputöö
Tehnotroonika õppekava

Juhendaja: Nooremteadur Erkki Jõgi

Tartu 2017

# LÜHIKOKKUVÕTE

<table>
<tr><td colspan="2">Eesti Maaülikool<br>Fr. R. Kreutzwaldi 1, Tartu 51014</td><td colspan="2">Bakalaureusetöö lühikokkuvõte</td></tr>
<tr><td colspan="2">Autor: Kaljo Valk</td><td colspan="2">Õppekava: Tehnotroonika</td></tr>
<tr><td colspan="4">Pealkiri: Automaatne kuiva kassitoidu jaotur</td></tr>
<tr><td>Lehekülgi: 135</td><td>Jooniseid: 19</td><td>Tabeleid: 4</td><td>Lisasid: 14</td></tr>
<tr><td colspan="4">Osakond: Tehnikakolledž<br>Uurimisvaldkond: Projekt<br>Juhendaja: Erkki Jõgi<br>Kaitsmiskoht ja -aasta: Tartu, 2017</td></tr>
<tr><td colspan="4">Antud töö eesmärk oli projekteerida automaatne kuiva kassitoidu jaotur. Seadme seadistamine pidi toimima üleni läbi veebiliidese, mitte vajama selleks internetiühendust ning oleks kasutajapoolsest operatsioonisüsteemi platvormist sõltumatu. Töö käigus tegi autor turuuuringu, püstitas lähteülesande, projekteeris seadme, samal ajal testis komponente, kirjutas neile tarkvaralist koodi ning ehitas osaliselt valmis prototüübi.</td></tr>
<tr><td colspan="4">Märksõnad: Kuivtoidu doseerimine, kasside toitmine, automaatne kuivtoidujaotur</td></tr>
</table>

# ABSTRACT

www.emu.ee
Eesti Maaülikool
Estonian University of Life Sciences

<table>
<tr><td colspan="2">Estonian University of Life Sciences<br>Kreutzwaldi 1, Tartu 51014</td><td colspan="2">Abstract of Master's / Bachelor's Thesis</td></tr>
<tr><td colspan="2">Author: Kaljo Valk</td><td colspan="2">Speciality: Technotronics</td></tr>
<tr><td colspan="4">Title: Automatic dry cat food dispenser</td></tr>
<tr><td>Pages: 135</td><td>Figures: 19</td><td>Tables: 4</td><td>Appendixes: 14</td></tr>
<tr><td colspan="4">Department: College of technology<br>Field of research: Project<br>Supervisor: Erkki Jõgi<br>Place and date: Tartu, 2017</td></tr>
<tr><td colspan="4">The aim of this thesis was to design an automatic dry cat food dispenser with the possibility of configuring the device entirely through a local web interface, without requiring an internet connection and to be independent of widely used operational systems. The thesis starts from studying the feeders market, setting the initial goal followed by the device design, testing of the physical components and the written software code. Finally, a partial prototype of the device was built.</td></tr>
<tr><td colspan="4">Keywords: Food dispensers, cat feeding, automated feeder</td></tr>
</table>

Valk K. An automatic pet food dispenser – Tartu, 2017. 132 pages, 4 tables, format A4. In Estonian.

# SISUKORD

# MÕISTED JA LÜHENDID

| | | |
|---|---|---|
| | - | |
| *24bitadcout1g* | - | ühele grammile vastav *ADC* väljund, |
| *AC/DC* | - | *Alternating Current/Direct Current*, vahelduvvool/alalisvool |
| *A* | - | *ADC* väljundile 1 vastav pinge väärtus, mV |
| *ADC* | - | Analoog-digitaalmuundur |
| *AES* | - | *Advanced Encryption Standard*, täiustatud krüpteerimis-standard |
| *adcm* | - | Ühele grammile vastav vaadeldava *ADC* väljund |
| *B* | - | maksimaalne *ADC* sisendile rakendatav pinge, V |
| *bps* | - | *Bits per second*, bitti sekundis |
| *C* | - | Kondensaator |
| *EEPROM* | - | *Electrically Erasable Programmable Read-Only Memory*, elektriliselt eemaldatav programmeeritav lugemiskaitstud mälu |
| *g* | - | gramm |
| *GPIO* | - | *General-purpose input/output*, üldotstarbeline sisend / väljund |
| *I* | - | Kogu kella voolutarve, mA |
| $I^2C$ | - | Jadaliides |
| *HostAP* | - | *Host Access Point, Wi-Fi* ligipääsu võimaldav tulipunkt |
| *Hotspot (Wi-Fi)* | - | Tulipunkt, *Wi-Fi* ligipääsu võimaldav seade |
| *IP* | - | *Internet Protocol address*, internetiaadress |
| *LCD* | - | *Liquid-Crystal-Display*, vedelkristallekraan |
| *LED* | - | *Light Emitting Diode*, valgusdiood |
| max24bitadcout | - | maksimaalne 24 *bit ADC* väljund 16777215 |
| *MCU* | - | *Microcontroller Unit*, mikrokontroller |
| *MOSFET* | - | *Metal–oxide–semiconductor field-effect transistor*, isoleeritud paisuga väljatransistor |
| *MSOP* | - | *Mini Small-Outline Package*, integraasleemide pakendi tüüp |

| | | |
|---|---|---|
| $Pm$ | - | Patarei mahtuvus milliamprit tunnis, mAh |
| *Q* | - | Rahaline kulu aastas |
| *R* | - | Takisti |
| *RTC* | - | *Real Time Clock*, reaalaja kell |
| *t* | - | Aeg tundides mille jooksul on patarei suuteline välja andma 3.1μA voolu pingel 3 V, h |
| *Ug* | - | tensoandurile rakendatava koormusele 1 g vastav vaadeldava ADC sisendi pinge, mV |
| $U_m$ | | *Vaadeldava ADC* väljundile 1 vastav pinge väärtus, mV |
| *UART* | - | *Universal Asynchronous Receiver/Transmitter*, jadaliides |
| *VDC* | - | *Volts of direct current*, alalispinge |
| *Wi-Fi* | - | *Wireless Local Area Networking*, traadita kohtvõrk |
| *WPA2* | - | *Wi-Fi Protected Access II*, traadita kohtvõrgu kaitstud ligipääs |

# SISSEJUHATUS

Läbi aegade on koduloomad imbunud inimellu ning vaikselt muutunud selle lahutamatuks osaks kõikjal maailmas. Alates 1970-st on koduloomade osakaal Ameerika Ühendriikides kolmekordistunud [1]. Lähtuvalt sellest on välja kujunenud vastav koduloomadele suunatud turuosa millesse on omanikud nõus oma raha paigutama. Näiteks kulutavad ameeriklased aastas üle viiekümne miljardi dollari koduloomadele ning üle poole Ameerika Ühendriikide elanikest omab kodus kas kassi või koera [1].

Siinjuures pööratakse suurt rõhku loomatoidule. Lemmikloomatoidu tootjad valmistavad mitmesuguseid tooteid, millel on vastavalt erinevad omadused ning koostis. Näiteks märg konservtoit võib sisaldada 80% vett ning ainult 20% kuivainet. Samas kuivtoidus võib olla 8% vett ja 92% kuivainet. Seega eristatakse kuiva ning märgtoitu. See töö keskendub kuivtoidule. Eestis müüb kuivtoite vähemalt 17 tootjat [2]. Seal hulgas Royal Canin, kes pakub 44 kuivtoidu toodet kassidele, omades kõige suuremat kassidele suunitletud toiduvalikut Eesti turul. Kuna kuivtoidu valik on suur ja Eesti piires kätte saadav käsitletakse töös Royal Canin-i poolt toodetavat kuivtoitu.

Käesoleva töö eesmärk oli projekteerida kasutajasõbralik ning kaasaegselt automatiseeritud seade mis annab ette kuivtoitu, mõeldud kodukassidele massiga kuni kuus kilogrammi kasutades seadmega suhtlemiseks tänapäevast tehnoloogiat nagu veebiserver, koduleht, traadita andmete ülekandmise võimalus *Hotspot (Wi-Fi).*

Eesmärgi saavutamiseks seati ülesanded:

1. Kirjeldada teoreetilist baasi;
2. Püstitada lähteülesanne;
3. Projekteerida seade;
4. Valmistada prototüüp.

# 1. TÖÖ TEOREETILINE BAAS JA SARNASTE SEADMETE ÜLEVAADE

## 1.1 Kuivtoidu valmistamine, maksumus, turg

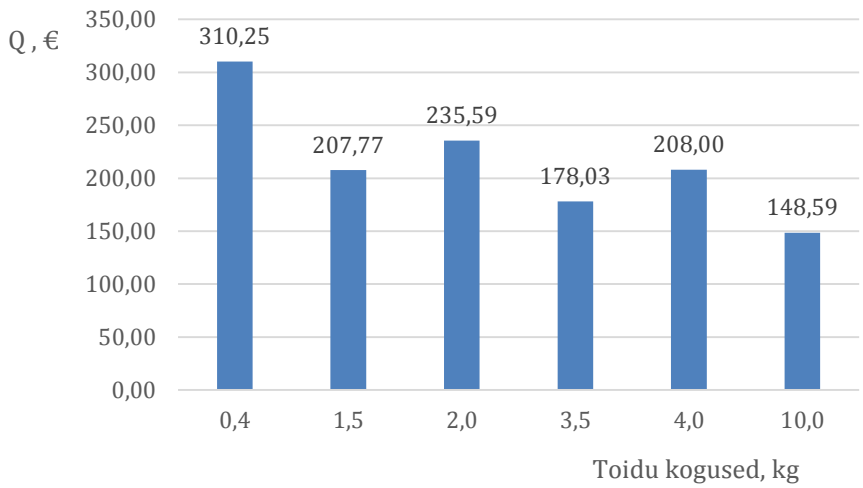
Kuivtoidu valmistamine toimub tavaliselt toidu keetmisega rõhu all ning järgnevalt toit kuivatatakse, pressitakse läbi vormide ning lõigatakse vajaliku suurusega tükkideks. Selline kuivatamise protsess teeb aga toidu loomale vähem maitsvaks. Selleks, et parandada toidu maitse omadusi lisatakse toidugraanulitele rasva, mis paraku rikneb kiiremini kui kuivaine. Kiirenevat toidu riknemist välditakse säilitusomadustega antioksüdantide lisamisega.

Kuna tootjaid ja pakutavaid toite on palju valis töö autor hindade võrdlemiseks tootja Royal Canin poolt pakutavad S*terilized* seeria tooted. Tootja poolt pakutavad kogused ja hinnad (autor külastas kassidele suunatud kuivtoitu müüvaid poode vahemikus 10.04.2017 - 16.04.2017 ja tõi välja keskmised pakendite hinnad mida onvõimalik Eestis osta) tabelis 1.1 ning kulud toidule aastas, mis on visualiseeritud joonisel 1.1.

Nagu selgus, suurem kogus toitu maksab vähem. Samas tuleb toidu ostul arvestada pakendi suurust ning toidu realiseerimisaega peale pakendi avamist.

**Tabel 1.1** Royal Canin S*terilized* seeria orienteeruvad müügil olevad kogused ja hinnad

| Pakendi mass, kg | Pakendi hind, € | Jätkuvus päevadeks | Kulud toidule aastas, € |
|---|---|---|---|
| 0,4 | 05,10 | 06 | 310,25 |
| 1,5 | 14,98 | 26 | 207,77 |
| 2,0 | 21,30 | 33 | 235,59 |
| 3,5 | 29,95 | 61 | 178,03 |
| 4,0 | 39,99 | 70 | 208,52 |
| 10,0 | 71,42 | 175 | 148,59 |




**Joonis 1.1.** Toidu maksumuse ja koguste võrdlus aasta lõpus

Lisaks mõõtis autor optimaalse pakendi suurust milleks oli 2 kg pakend kuna selline pakend säilib 33 päeva avamisest (kirjas pakendil). Seega suurema pakendi ost ei ole ühe looma jaoks põhjendatud. Lähtudes tabelist 1.2, kass kaaluga 6 kg ei suuda kogu pakendis olevat toitu enne säilivusaja lõppu ära süüa.

Optimaalse pakendi suuruseks sai autor 90 x 170 x 270 mm ning arvutas pakendi ruumala korrutades omavahel pakendi pikkuse, laiuse ja kõrguse. Saadud tulemuse arvestas liitrites ja sai pakendi mahutavuseks 4,131 liitrit.

## 1.2 Kuivtoidu säilitamine ja hügieen

Kuivtoidu säilivuse ja tingimuste kohase info toob välja tootja oma veebilehel, näiteks Royal Canin, küsimuste ja vastuste all [3]. Vastava peatüki all on kirjas, et Royal Canin kuivtoit säilitab oma maitseomadused 1,5 kuni 2 kuud peale pakendi avamist. Toit tuleb säilitada soovitatavalt originaalpakendis. Säilituskoht peab kuiv ja hästi kaitstud temperatuurimuutuste, otsese päikesevalguse ning parasiitide eest. Kui säilitustemperatuur on pakendile märgitud, tuleb sellest lähtuda, vastasel juhul ei ole erilisi nõudeid temperatuurile. Pakend tuleb alati korralikult sulgeda, eemaldades sellest võimalikult palju liigset õhku. Lisaks soovitatakse valida pakendeid vastavalt loomade suurusele ja arvule.

Näiteks 10 kg suurune pakend võib osutuda liiga suureks üksiklooma jaoks ning võib tähendada pikemat säilitusaega mis ületab kahte kuud.

Toit ei tohiks olla kausis kauem kui 24 tundi. Vastasel juhul kaotab ta oma maitse ja lõhnaomadused. Pinnale ladestub tolm ning toidu pinnal olev rasv jõuab biokeemiliselt rikneda ehk oksüdeeruda. See on kõige sagedam põhjus, miks kassid keelduvad sööma kausi põhjas olevat toitu [4]. Seepärast on parem toitu anda ette väiksemate portsjonite kaupa päeva jooksul kui anda ette kohe terve päevase normi. Toidu riknemise ennetamiseks soovitatakse regulaarselt toidukaussi pesta puhta veega, see aitab vältida hallituse kasvu, putukate ja bakterite paljunemist toidus ning aitab takistada toidus olevate õlide poolt tingitud toidu halvaks minemist.

## 1.3 Kasside toitmise metoodika

Kuivtoidu korral soovitab toidu annustamise kogust kuivaine tootja andmeleht. Info on leitav vastava söögi pakendil, kus annustamise kogused tabelina kirjas on. Lähtuvalt looma vanusest, kaalust, tervislikust seisundist, elustiilist, tõust, füüsilisest seisundist ning võimalikust haigusest. Kindlasti peab kuivtoidu kõrval olema nõu veega. Vastavalt kassipidaja käsiraamatule peaks terve kass sööma 2 korda päevas, kassipojad aga 3-4 korda [4].

Tuleb mainida, et iga loom on isemoodi, mõni on rohkem, teine vähem aktiivne ning tegeliku toidu koguse ning söögikordade üle otsustab lõppkokkuvõttes siiski looma omanik. Tabelid pakendite tagakülgedel on vaid orientiiriks. Näiteks valitud toiduseeria pakendite tagaküljel olevad keskmised soovituslikud päevased annused, tabel 1.2.

**Tabel 1.2.** Tootja soovituslikud päevased annused steriliseeritud kassile, kg

| Looma mass | 3 | 4 | 5 | 6 |
|---|---|---|---|---|
| Annus ideaalse massiga loomale | 0,049 | 0,060 | 0,070 | 0,079 |
| Annus ülekaalulisele loomale | 0,039 | 0,048 | 0,056 | 0,063 |

Kuigi soovitatavad annused on ette antud, esineb siiski väga palju probleeme koduloomade ülekaalulisusega. Vastavalt koduloomade rasvumise vähendamise ühingule on üle poole, 59% kasse Ameerikas ülekaalulised. Põhjus on üks, loomadele antakse ette liiga palju sööki [5].

Kuna töö autor valis Royal Canin poolt pakutavad S*terilized* seeria tooted on oluline mainida muutusi, mis tulevad esile peale kassi steriliseerimist [6]:

1. Täiskasvanud kassidel aeglustub ainevahetus, mis võib osutuda rasvumise põhjuseks ebaõige toitumise korral;
2. Looma energeetilised vajadused muutuvad väiksemaks, seejuures söögiisu kasvab 18-26% võrra.

Oluline on kinni pidada kindlast söögigraafikust andes kassile süüa iga päev samal ajal, et loom ei läheks stressiseisundisse ning et tema ainevahetus stabiliseeruks. Selleks tuleb leida söögi jaoks kindel koht, kus loom tunneb end turvaliselt ja mugavalt. Kuna kass võib nii üle kui liiga vähe süüa, võib see olla haigusnähu märk. Sellisel juhul tuleb konsulteerida loomaarstiga.

## 1.4 Turuuring

Töö autor analüüsis ja võrdles hetkel maailmaturul 8-t kassidele mõeldud kuivtoidu jaoturi (LISA A, tabel 1) [7]:

1. *Petsafe simply feed;*
2. *Csf-3 super feeder;*
3. *Pet feedster;*
4. *Lusmo automatic pet feeder;*
5. *Crown majestic (diamond series v 3);*
6. *Petnet smart feeder;*
7. *Cat mate c3000;*
8. *Wireless whiskers pet feeder;*

Võrreldes olemasolevaid seadmeid selgusid mitmed probleemid. Uuringu käigus selgus, et enamus tooteid jäid mahutavuse poole pealt alla 1,18 L, kusjuures optimaalne suurusjärk sellise seadme jaoks on 4,131 L, mis sai 1.1 peatüki lõpus välja arvutatud. *Petsafe simply feed* seadme mahutavus oli näiteks 6 L mis on tunduvalt suurem optimaalsest.

Oluliseks probleemiks oli ka programmi kustumine voolukatkestuse korral. Enamik seadmeid olid võimelised küll programmi patarei toitel säilitama kuid patarei vahetusel programm kustus. Seadistuse suutis säilitada vaid *Pet feedster*, v.a kell.

Kasutajamugavuse poole pealt pakkus vaid 1 seade nutitelefoni tuge (*Petnet smart feeder*) kuid see piirdus vaid vaid *iOS* 8 – või uuema *iOS* operatsioonisüsteemi versiooniga. Ka ei olnud võimalik kasutada selleks veebilehitsejat, ainult *iOS* rakendust. Seadet ei olnud võimalik algseadistada ega hiljemalt seadistust muuta ilma internetita. *Csf-3 super feeder* pakkus siin unikaalset seadistamisvõimalust kruvikeerajaga seadme alt, tegemaks seadistamisprotsessi võimalikult ebamugavaks ning keeruliseks.

Ükski uuritud seadmetest ei olnud kooskõlas autori visiooniga kasutajasõbralikust seadmest.

# 2. LÄHTEÜLESANDE PÜSTITAMINE

## 2.1 Kontseptsioon

Enne seadme projekteerimist tuli selgusele jõuda mis on tegelik probleem mida lahendada tuleb ning miks. Selleks tuli vastata muuhulgas järgmistele küsimustele:

1. Seadme kontseptsioon;
2. Kes seda kasutama hakkab?
3. Miks seda kasutama hakatakse?
4. Kus kohas seda kasutama hakatakse, millistes tingimustes?
5. Millal seda kasutama hakatakse?
6. Kuidas seda kasutama hakatakse?

Kuna oli oluline leida seadmele sobivad parameetrid siis eelnevaid küsimusi oli tähtis lähteülesande püstitamisel silmas pidada. Vastasel juhul tekkinuks suur risk toote väljaarendamise aja pikenemises, uute omaduste juurde lisamise või olemasolevate muutmise korral:

1. Seadmeks on kuivtoidu etteandja mehhanism mis on mõeldud kodu kassidele massiga kuni 6 kg;
2. Seadet hakkavad kasutama kassid, jaotur on mõeldud ühele kassile, samas seadet seadistama, täitma ja puhastama hakkavad inimesed. Nendeks võivad olla lapsed, täiskasvanud. Vastavast uuringust [8], selgus et kõrge surma riskiga kassiomanikud (vanurid näiteks) elavad kauem. Seega on igati põhjendatud vanuritel kodukassi olemasolu ja sellega koos projekteeritava jaoturi potentsiaalne kasutamine. Lisaks vanuritele võttis töö autor arvesse ka Eestis elavaid piiratud liikumisvõimega inimesi. Tervisestatistika ja terviseuuringute andmebaasi andmete järgi [9] on seisuga 2016 aastal, 10,7% Eesti elanikkonnast piiratud liikumisvõimega, seega loob seade erilist väärtust ka sellistele inimeste kuna toidu lisamist ning seadme hooldust saab teha kord pooleteise kuu järel hooldaja, ülejäänud aja töötab seade autonoomselt;
3. Seadet hakatakse kasutama järgmistel eesmärkidel:

3.1. Et anda sööki ette aegadel, mil loomaomanik ise seda teha ei saa või tal on seda ebamugav teha, näiteks hommikuti;
3.2. Anda ette sööki kindlatel aegadel, et ennetada looma stressi;
3.3. Looma massi langetamise eesmärgil, juhul kui loom on ülekaaluline;
3.4. Seade säilitab toidu looma jaoks värskena ligikaudu poolteist kuud, seega toidu säilitamise eesmärgil;
3.5. Selleks, et ostetud toidupakendit ei peaks looma eest peitma ega selle jaoks eraldi ladustamise ruumi looma. Kogu ostetud pakendi sisu säilitatakse jaoturis olevas, selleks ette nähtud konteineris;
3.6. Loom võib püüda ise käpaga jaoturi avausest toitu kätte saada või käituda agressiivselt ning jaoturi füüsiliselt mõjutada. Seega tuleb disainida seade muuseas selleks, et loom ise toitu kätte ei saaks.

4. Seadet hakatakse kasutama ruumi sisetingimustes temperatuuril ligikaudu 22 kraadi;
5. Eeldades, et loom nõuab süüa iga päev ning teha ta võib seda näiteks vara hommikul ehk siis kui loomaomanik veel magab või näiteks öösel. Seega hakatakse seadet kasutama ööpäevaringselt kassi poolt. Inimene seadistab seadet suure tõenäosusega vaid 1 kord, kuna seaded ei kustu voolu katkedes. Kella õigeks seadmine toimub kellapatarei vahetusel. Seadme hooldus (nõude, konteineri puhastus) kord pooleteise kuu järel;
6. Kasutaja hakkab jaoturit konfigureerima seadmega, mis suudab ühenduda *Hotspot (Wi-Fi)* võrku ning omab veebilehitsejat, näiteks nutitelefoni või arvutit. Veebiserver kuvab kasutajale antud lokaalsel veebiaadressil veebilehe, kus saab kasutaja sisestada hetke kellaaja, mitu korda soovib looma toita, millistel aegadel, kui suurte portsjonite kaupa ning lubab seadme sätted salvestada. Loom hakkab sööma toitu ja jooma vett selleks ettenähtud nõudest mis on eraldiseisvad jaoturist ning on kasutaja enda poolt valitavad ja ei kuulu seadmekomplekti hulka.

## 2.2 Nõuete määratlemine

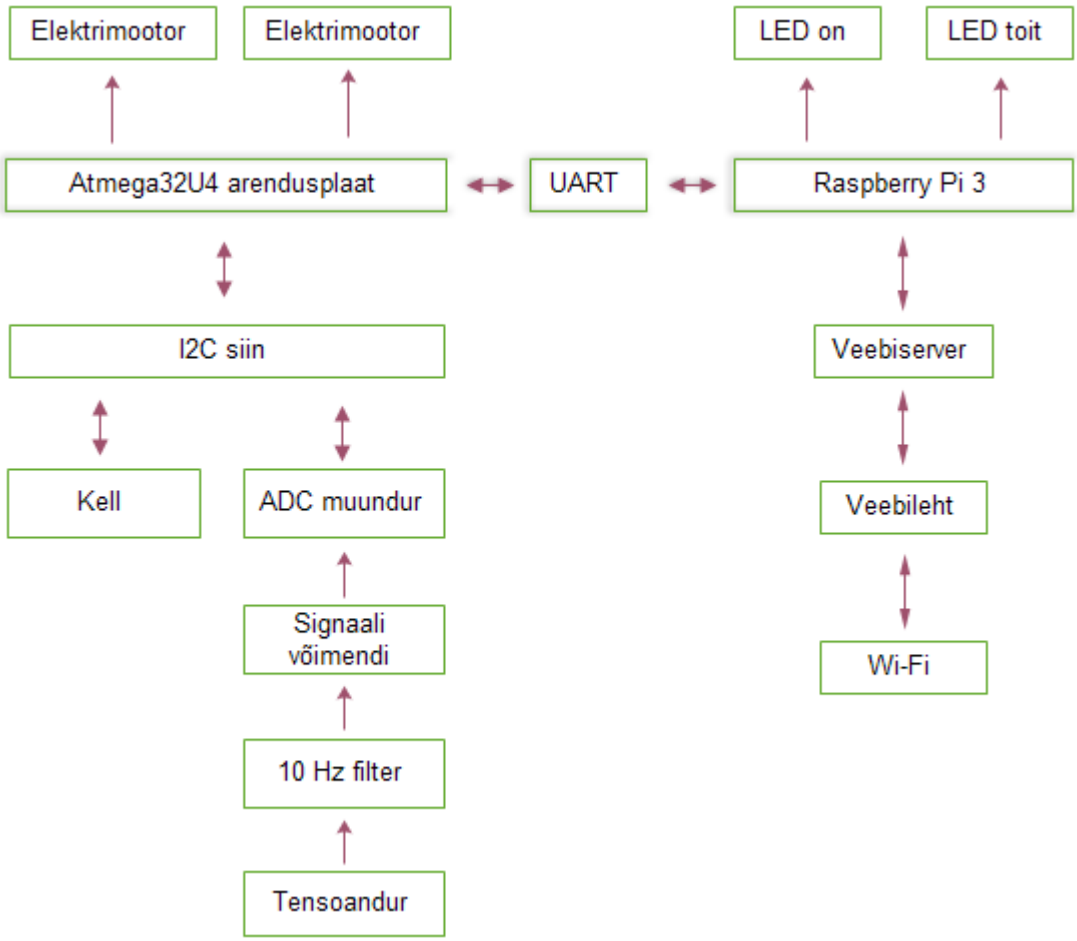
Kõigepealt pani töö autor paika elementaarsed nõuded nõuetele millest edaspidi lähtuda:

1. Üheselt ja lihtsasti mõistetavad;
2. Arvestavad tehnoloogia piiranguid ning reaalset maailma;
3. Selge idee esitamine.

Paika tuli panna ka tegevusplaan juhuks kui paikapandud nõuded ei olnud projekti algul õigesti struktureeritud või mingil põhjusel projekti jooksul muutuvad. Näiteks võis selguda et mõninga funktsionaalsuse implementeerimine võis osutuda liiga keeruliseks või kulukaks. Lähtuvalt sellest otsustas töö autor kasutada agiilset lähenemist ja valideerida projekti igas võimaliku faasis kasutades prototüüpimist ning modulaarset lähenemist eesmärgile. Nõuded said paika pandud vastavalt lähteülesande püsitusele peatükis 2.1, siinjuures jaotas autor nõuded eraldi funktsionaalseteks ja mittefunktsionaalseteks.

## 2.3 Seadme arhitektuuri visualiseerimine

Visualiseerimaks modulaarse süsteemi tööd lõi autor moodulite omavahelist interaktsiooni kirjeldava skeemi, joonis 2.1, kus nooled näitavad signaalide ja andmete liikumise suunda.




**Joonis 2.1.** Lihtsustatud seadmete omavahelist suhtlust visualiseeriv skeem.

Seade koosneb kahest põhimoodulist, *Atmega32U4* mikrokontrolleril baseeruvast arendusplaadist ning Raspberry Pi seadmest, mis suhtlevad omavahel kasutades *UART* liidest. Raspberry Pi peamiseks ülesandeks on indikeerida kastajale seadme valmisolekust andmeid vastu võtma, hallata suhtlust *hotspot-i*, kasutaja, veebiserveri ja veebilehe vahel. Vastavalt kasutajalt saadud andmetele edastada vajalikud seadme sätted arendusplaadile, mis omakorda haldab suhtlust läbi $I^2C$ siini enda ning reaalajakella ja analoog-digitaalmuunduri vahel ning kontrollib elektromootorite tööd.

## 2.4 Funktsionaalsed nõuded

Funktsionaalsed nõuded kirjeldavad kasutaja seisukohalt süsteemi käitumist või kindlat funktsiooni, näiteks mida süsteem tegelikult teeb või ei tee. Seade koosneb moodulitest, mis täidavad kindlaid funktsioone:

**Seade kui tervik:**

1. Seade doseerib kuivtoitu etteantud aegadel;
2. Annab märku kui konteineris saab toit otsa;
3. Säilitab toitu kuni 33 päeva;
4. Lubab seadme seadistamist juhtmevabalt;
5. Lubab doseerivate koguste seadistamist;
6. Lubab seadistada annustamise sageduse päevas;
7. Töötab 5 V toiteadapteriga maksimaalse väljundvooluga 2,5 A.

**Reaalaja kella moodul:**

1. Võtab vastu uued kellaseaded;
2. Säilitab reaalajas kella;
3. Edastab päringu peale *Atmega32U4* mikrokontrollerile tunnid, minutid, sekundid.

**Analoog-digitaalmuunduri moodul:**

1. Võtab vastu mooduli sätteid;
2. Konverteerib tensoandurilt analoogsignaali digitaalseks;
3. Edastab saadud tulemused küsimise peale *Atmega32U4* mikrokontrollerile;

**Elektrimootorid:**

1. Esimese ajami ülesandeks on pöörata trummlit;
2. Teise ajami ülesandeks on avada ja sulgeda kaalumiskausi klappi;

## 2.5 Mittefunktsionaalsed nõuded

Mittefunktsionaalsed nõuded määravad teenuse osutamise kvaliteeti ehk täpsustavad nõuete kriteeriume nagu näiteks seadme tööd, erinevalt funktsionaalsetest nõuetest. Kokkuvõtvalt defineerivad mittefunktsionaalsed nõuded seda, milline peab süsteem kasutaja seisukohalt olema.

**Seade kui tervik:**

1. Seadme välised gabariidid. Pikkus, Laius, kõrgus (250 x 500 x 100) mm;
2. Toidukonteineri mahutavus 90 x 170 x 270 mm (4,131 L);
3. Kaitse, et loom ei saaks ise käpaga toitu kätte;
4. Kaitse, et loom ei saaks süüa enne kui kogu toit on kausis;
5. Kaas ei tule lahti kui seade ümber läheb;
6. Toit ja vesi eraldi animates;
7. Looma söögikauss mahutab kaks korda rohkem kui 0,080 kg kuivtoitu, mis on suurim portsjoni suurus, et kass söömise käigus ei lükkaks ninaga graanuleid üle kausi ääre;
8. Süsteemi seaded ega kell ei kustu voolukatkestuse korral;
9. Heli märguanne kassile, et söök on ette antud.

***Raspberry Pi:***

1. Annab kasutajale märku, et veebiserver on aktiviseerunud (pannes *LED-i* põlema);
2. Loob *Wi-Fi Hotspot-i* nimega *Catspot* ja turvab *selle WPA2* krüpteeritud salasõnaga;
3. Käitab veebiserverit, mis kuvab kasutajale veebilehe etteantud aadressil;
4. Kasutab *UART* liidest konfiguratsiooniandmete edastamiseks ja vastu võtmiseks;
5. Võtab *Atmega32U4* mikrokontrollerilt vastu põhiseaded;
6. Kuvab seaded veebilehel:
    6.1 Tunnid, minutid, sekundid;
    6.2 Mitu korda päevas sööta;
    6.3 Millistel kellaaegadel sööta;
    6.4 Kui palju toitu ette anda ühe korraga.

7. Võtab kasutajalt vastu uued seaded ja saadab *Atmega32U4* mikrokontrollerile salvestamiseks;

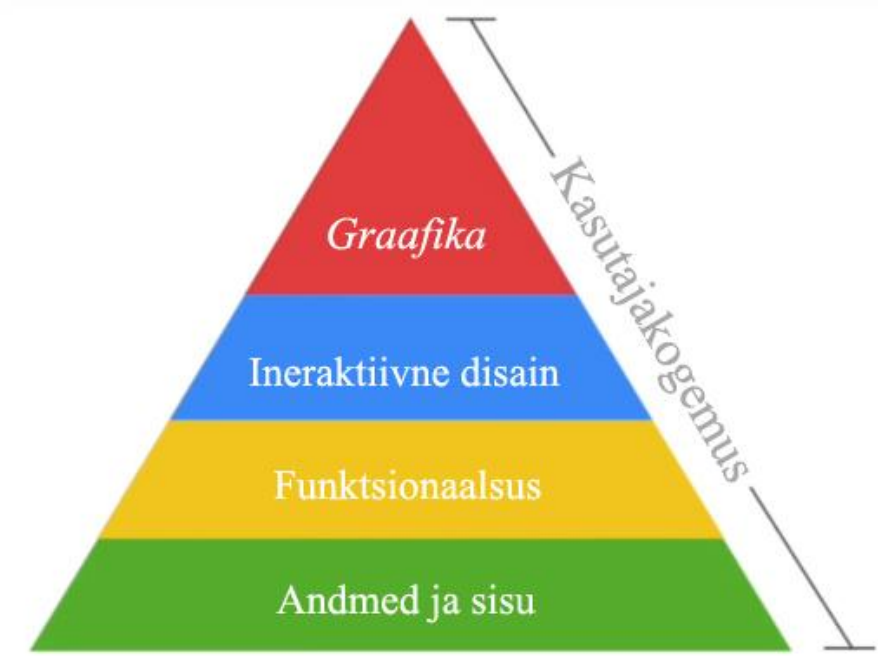
**Kasutajaliides**

Ütlemaks lahti vananenud kontseptist, kus seadme konfigureerimiseks pidid olema füüsilised lülitid koos tagasiside ekraaniga, sai otsustatud kasutada kaasaja tehnoloogiaid ning projekteerida seade ilma ühegi füüsilise lülitita, et luua kasutajaliides mida on lihtne ja enesestmõistetav kasutada, ehk panna rõhku kasutajakogemusele, eesmärgiga viia kogu seadme seadistamise ja kasutajavahelise suhtluse veebilehe vahendusele, millele rakendada järgnevaid nõudeid:

1. Lihtne ja enesestmõistetav keelekasutus;
2. Kuvada infot kasutaja keeles;
3. Minimiseerida mälu kasutust (inimesed jätavad infot halvasti meelde);
4. Veateated peavad olema konstruktiivsed e. kasutajale mõistetavad (miks ja mida kasutaja saab teha, et edasi liikuda, mitte jääda veateate juurde kinni);
5. Meeldetuletused (inimese jaoks on tunduvalt lihtsam meelde tuletada varasemat infot. Andes ette selge hulk valikuid on inimene võimeline ära tundma valikute seast talle vajalikku informatsiooni, selle asemel kui lasta tal täita tühja lahtrit);
6. Öelda inimesele mida edasi teha. Näiteks “sinu seaded on salvestatud, võid seadme välja lülitada”;
7. Anda tagasisidet kasutajale kui miskit valesti või õigesti läheb või kui kaua peab veel ootama, kas piisab kohvipausist või läheb tunde;
8. Püüda ette aimata vigu ja neid vältida;
9. Mitte lubada kasutajal sisestada andmeid valesti;

Loetelust lähtuvalt tuli paika panna ka kasutaja kogemuse põhimõtted vastavalt *Janne Jul Jensen* soovitustele *Trifork* veebiarendajate konverentsil 2012 ja 2015 aastal, tema *User Interface (UX) Techniques* ettekandes [10]. Selleks tuli uurida kasutaja vajaduste ehk kasutajakogemuse püramiidi (joonis 2.2). Põhjuseks on kas andmete puudumine või õigsus, kuna see võib anda kasutajale halva kasutajakogemuse. Kui valitud funktsionaalsus oleks

olnud liiga suur või väike, oleks see jällegi halb kasutajakogemus. Kui kasutaja ei saanuks etteantud andmetes navigeerida, jälle halb kasutajakogemus ning kui veebileht näeks kohutav välja – kõik ülevalpool nimetatu mõjutab kasutajakogemust negatiivselt.




**Joonis 2.2.** Kasutajakogemuse püramiid [10]

**Andmed ja sisu –** juhul kui on midagi vajalikku kasutajale näidata. Kui sisu pole, siis ei oma tähtsust kui teha multifunktsionaalne ja ilus kasutajaliides või navigatsioon.

**Funktsionaalsus –** Paraku on sisu ja andmeid tänapäeval üleküllluses, ka selle seadme puhul ning kui kasutajale lihtsalt kogu info ette visata siis kasutaja ei tea enam mida teha. Seepärast on tarvis kõike filtreerida ning ettevaatlikult valida, millal ja millist infot kasutajale kuvada. Mitte palju, mitte vähe, just piisavas koguses.

**Interaktiivne disain –** Peale seda kui on otsustatud mida ja millal, tuleb selgusele jõuda kuidas seda kasutajale näidata ning kuidas kasutaja seadmega suhtlema hakkab. Selle realiseerimiseks on lõpmatu hulk võimalikke mooduseid, tuleb valida sobilik.

**Graafika –** Alles siis kui kõik eelnev on paigas saab mõelda graafilise disaini peale.

# 3. SEADME PROJEKTEERIMINE

## 3.1 Arendusplaat

Kuna tegu on prototüüpprojektiga, mille käigus võis palju muutuda otsustas autor luua tööriista, mis võimaldanuks muutusi projekti kiiresti sisse tuua. Selleks kavandas autor arendusplaadi (LISA B), kasutades selleks *Eagle* [23] tarkvara mille hulka kuulusid *UART* ja $I^2C$ liidesed, reaalajakell, M*icro-USB* pesad arendusplaadi programmeerimiseks, *UART* suhtluseks ning väljaviigud mikrokontrolleri jalgadelt. Lisaks sellele lülitid plaadi programmeerimiseks tarbeks ja testimiseks ning erinevad *LED* indikaatorid vigade indikeerimiseks (näiteks indikaatorid mis näitavad kas integraalskeemi väljaviigud on pingestatud, kas *I2C* ja *UART* liidestes liiguvad andmed). Seejuures *I2C* liidese kasutus sai disainitud nii, et oleks võimalik valida, kas kasutada plaadile integreeritud reaalajakella või siis muud välist *I2C* seadet. Sellel oli 2 põhjust, esiteks $I^2C$ liides võimaldab ühendada endaga mitmeid seadmeid, teiseks võivad seadmete aadressid konflikti sattuda (kahel seadmel võib olla sama aadress), sellisel juhul võib olla tarvilik plaadile integreeritud seadme $I^2C$ liidese küljest lahti ühendamine. Samuti oluline viikude disaini aspekt oli toite radade disain kus ette teades, et seadet tuleb programmeerida ühe *USB* liidesega ning kommunikatsioon läbi *UART-i* hakkab toimima mööda teist *USB* liidest oli igati loogiline disainida toiteühendused nii, et seade saaks toidet läbi mõlema *USB* juhtme. Selle tulemusena on võimalik arendusplaadi toitmiseks pingega piirduda vaid ühe *USB* juhtmega peale seda kui seadmele on kood peale programmeeritud.

Arendusplaadi tööülesanded:

1. Suhtleb läbi *UART* liidese *Raspberry Pi*-ga;
2. Saadab seadeid *Paspberry Pi*-le ja võtab talt neid vastu;
3. Suhtleb läbi $I^2C$ liidese kella ja *ADC* mooduliga;
4. Saadab kellale seadeid ja kellaaega ja võtab kellaaja vastu;
5. Saadab *ADC* moodulile seadeid ja võtab vastu digitaalset mõõtetulemust;
6. Juhib elektrimootorit;
7. Avab söögiava klappi;
8. Hoiab seadeid püsimälus, *EEPROM*-is;

Arvesse võttes eelnevat ning töö autori kogemust *Atmega* mikrokontrolleritega sai arendusplaadi tuumaks valitud *Atmega32U4* [13], hinnaga 11,29 €.

## 3.2 Korpuse ja seadme sisedetailide projekteerimine

Lähtuvalt toidu säilitamise nõuetest peab toudukonteineriga ühilduv süsteem olema hermeetiline. Selles projektis ei ole rõhutud täielikule hermeetilisusele. Sellest lähtuvalt sai otsustatud disainida komponendid nii, et hermeetilisust oleks tulevikus võimalikult lihtne täiendada. Vastavalt sellele kavandas autor söögi konteinerile hermeetilise sulguri (LISA I), kuubi ja lehtri ühendused (LISA F ja G), jättes kuubil aluse lahti. Järgnevalt kavandati kuup, mis tsentreerub automaatselt vastavalt lehtri positsioonile ning asetses vahelülil (LISA J) lahtiselt, et seda oleks võimalik hõlpsasti seadmest hooldamiseks eemaldada. Lehtrist kaalumiskaussi doseeriv trummel (LISA G) hoiab toitu kinni elastse takistuse mõjul. Trumli pöörlemise tagajärjel kummist takistus deformeerub ning toit kukub kaalumiskaussi (LISA H), mis asetseb lahtiselt tensoanduri ülemise kinnituse peal (LISA E).

Kaalumiskausi sulgurklapp mille füüsiline paiknemine on otseselt seotud elektrimootori omaduste- ja gabariitidega. Komponendi mõõtmed, kavandamine ja seadmes paiknemine tuli kindlaks teha viies läbi reaalseid katseid milleni aga töö käigus ei jõutud. Seetõttu jäi klapi projekteerimine antud töö fookusest välja kuid selle lisamist tulevikus oli silmas peetud ning vastavalt sellele jäetud korpuses ruumi komponendi lisamiseks.

Kaalumiskausile projekteeriti ava mille eesmärk oli kausi tühjendamine sujuvalt gravitatsioonijõu mõjul. Kuna toit valgub avast koheselt välja, siis oli tarviklik lisada elektriliselt avatav sulgur. Oluline oli disainida kaalumiskauss selliselt, et seda oleks võimalik kergesti seadmest eemaldada ning tagasi panna täpselt samasse kohta. Selle tarbeks sai disainitud kaalumiskausile vastav põhi, mis tsentreerus automaatselt tensoanduri ülemisele kinnitusele ning selle paigaldamine sai toimuda ainult ühel viisil, avaga õiges suunas.

Seadme korpuse küljed ja toed said disainitud vineerist (LISA J).

## 3.3 Kasutajaliidese disain

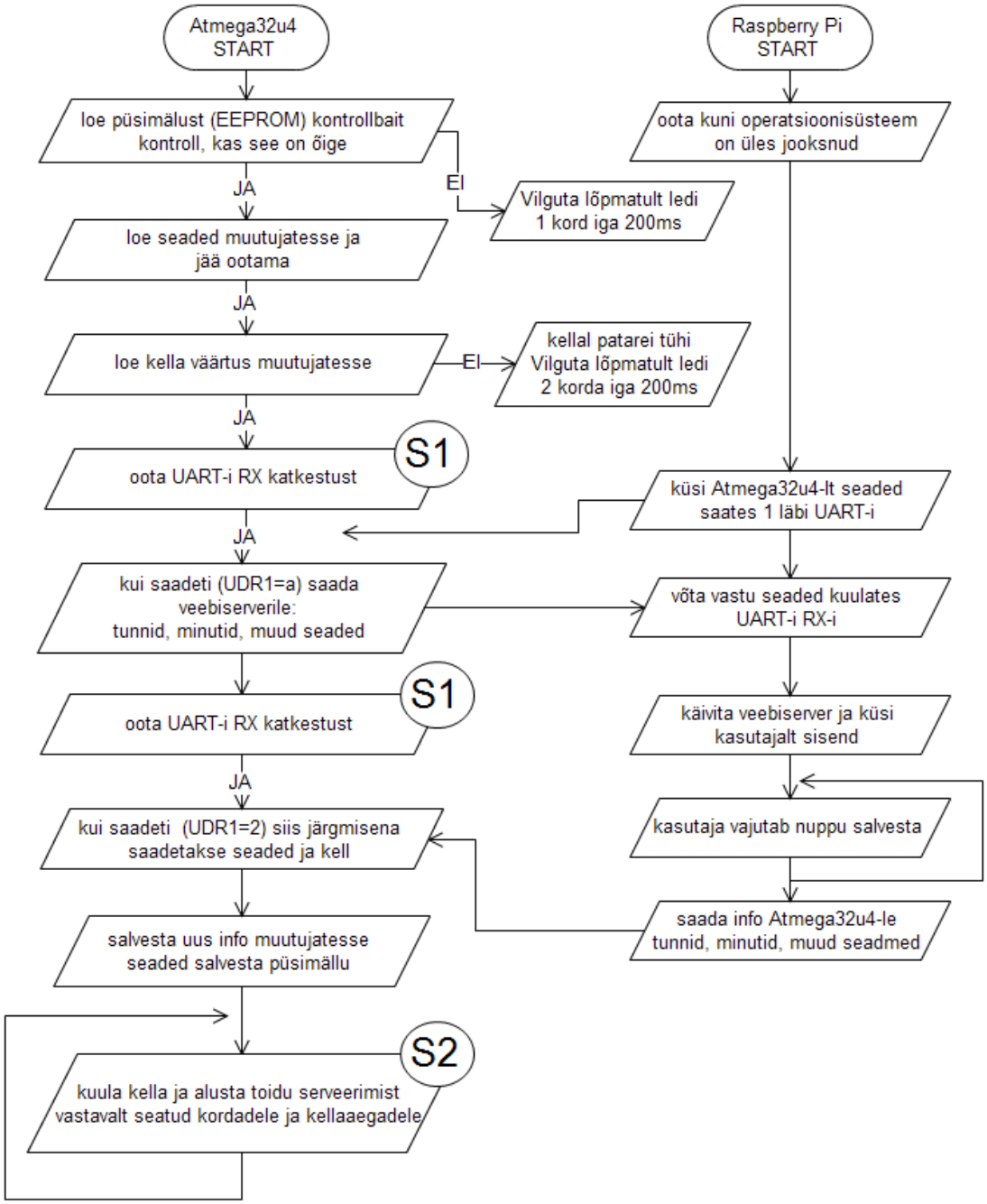
Kasutajaliidese loomisel sai lähtutud kasutajaliidese nõuetest (peatükk 2.4). Selleks otsustas autor võtta kasutusele *Raspberry Pi 3* mudeli B, millele on *WI-Fi* moodul sisse ehitatud. Operatsioonisüsteemiks valiti *Rasbian* 4.4. Seade sai valitud *hotspot-i* loomiseks ning veebiserveri käitamiseks.

*Hotspot-i* loomiseks planeeriti kasutada *hostapd* [27] tarkvara moodulit, konfigureerida järgneva õpetuse kohaselt [11], turvata ühendus *WPA2* võtmega ja keelata kasutajal ligipääs internetti. Nii on kasutajal võimalik ühenduda seadme poolt tekitatud kohtvõrguga ning külastada veebiserveris töötavat veebilehte kusjuures internetiühendust selle jaoks ei ole tarvis.

Veebiserver oli plaanis panna tööle *Python-is* kirjutatud *Flask* veebiraamistikus. Selle tarbeks tuli installeerida *Flask* tarkvarapakett ning *WTForms* alampakett, mis võimaldab kasutada vorme veebilehel. Kasutajakogemusele mõeldes oli plaanis lisada indikaator *LED Raspberry Pi GPIO 3 ja 6* väljaviigu vahele näitamaks, millal *WI-FI hotspot* aktiveerub. Nii *hotspot* kui veebiserver aktiveeruvad seadme vooluvõrku ühendamisel.

## 3.4 Arendusplaadi ja Raspberry Pi vaheline kommunikatsioon

Arendusplaat ja *Raspberry Pi* suhtlevad omavahel kasutades *UART* liidest. Liidese konfiguratsioon sai paika pandud mittefunktsionaalsete tarkvara nõuete (ptk 2.5) näol. Seadmetevaheline suhtlus on visualiseeritud joonisel 3.2 kus nooled näitavad andmete liikumise suunda ning *S1* ja *S2* tähised hetkest programmi oleku seisundit.




**Joonis 3.2.** Arendusplaadi ja *Raspberry Pi* vahelise kommunikatsiooni põhimõtteskeem.

Skeemil ei ole märgitud kuidas otseselt saadakse olekutesse *S1* ja *S2* kuna programm töötab katkestuste põhimõttel. See tähendab, et ettemääratlemata ajahetkel võib tekkida katkestus, mis katkestab kogu programmi töö, peale mida täidetakse katkestuserutiin ning programm jätkab sealt, kus pooleli jäi.

## 3.5 Tensoanduri valik

Tensoanduri valikul lähtus autor neljast parameetrist: hind, võimekus, varutegur ning tarneaeg. Kuna komponenti ei müüda Eestis, tuli see tellida välismaalt. See aga eeldab pikka tarneaega. Arvestades toiduannuseid, mis on ligikaudu 0,1 kg 6 kilogrammise looma kohta ning varutegurit ehk kausi enda massi mis võib olla ligikaudu 0,25 kg ning anduri külge ühendatavate paneelide masse, ligikaudu 0,15 kg ning enda anduri enda maksumust jõudis töö autor järelduseni, et on tarvis andurit mille võimekuse mõõtepiirkond on vähemalt 0,5 kg, et rakendatav jõud ei ületaks andurile maksimaalset ettenähtud koormust. Lähim nõuetele vastav andur oli 0,5 kg ning maksis ~8 € (seisuga 11.05.2017 müügist eemaldatud). Samas mõõtepiirkonnaga 1 kg anduri maksumus oli ~2,50 € [12]. Valitud sai soodsam 1 kg andur parameetritega märgitud tabelis 3.1.

Ligikaudne maksimaalne koormus mida autor mõõta soovis oli 0,5 kg, seega rakendades 0,5 kg andurile mõõtepiirkonnaga 0,5 kg on analoogsignaali konverteerimine digitaalsele kujule 50% võrra täpsem võrreldes 1 kg mõõtepiirkonnaga anduriga millele rakendatakse samuti 0,5 kg. Autor lahendas probleemi tehes vajalikud kalkulatsioonid (ptk 3.5) ning valis analoog-digitaalmuunduri (*ADC)* mille resolutsioon oli antud ülesande lahendamiseks piisav.

**Tabel 3.1.** Valitud tensoanduri parameetrid [12]

| |
|---|
| Maksimaalne koormus: 1 kg |
| Väljund: 1,0 mV/V ± 0,15 mV/V |
| Null väljund: ± 0,1 mV/V |
| Sisend, Maa: *Red+ (power), Black-(power)* |
| Väljundid: *Green+(signal), White-(signal)* |
| Soovitatud toitepinge: 3 - 12 *VDC* |
| Maksimum toitepinge: 15 *VDC* |
| Sisendi näivtakistus: 1115±10%Ω |
| Väljundi näivtakistus: 1000±10%Ω |

Tensoanduri kinnitamiseks korpuse külge sai projekteeritud tensoanduri alumine kinnitus (LISA E) ning ülemine kinnitus, mille peal asetseb kaalumiskauss (LISA E).

## 3.6 Analoog-digitaalmuunduri valik

Vastavalt *Atmega32U4* andmelehele [13] on vaadeldavasse mikrokontrollerisse integreeritud *10 bit* analoog-digitaalmuundur. Autor plaanis muunduri sisendile rakendada maksimaalse väärtusena 5 V. Kuna *10 bitise* muunduri väljundiks on number vahemikus 0-1023 ($2^{10}$- 1= 1023), jagas autor 5 V 1023 võrdseks osaks ning sai ühe osa suuruseks ümardatult 4,8876 mV, mis võrdub muunduri väljundi ühikuga 1, ehk *10 bit ADC* 1 ühik = 4,8876 mV (valem 3.1).

$$U_m = \frac{U_s}{1023} = \frac{5}{1023} \approx 4{,}8876 \text{ mV}, \tag{3.1}$$

kus $U_m$ on vaadeldava *ADC* väljundile 1 vastav pinge väärtus, mV;

$U_s$ – maksimaalne vaadeldava *ADC* sisendile rakendatav pinge, V.

Võttes arvesse tensoanduri valikut, milleks sai andur maksimaalse koormusega 1 kg ning potentsiaalset mõõtetäpsust 1 g, teisendas autor 1 kg grammideks ning jagas 5 V 1000-s võrdseks osaks ning sai teada, et 1 g võrdub 5 mV. Seega rakendades valitud tensoandurile koormust 1 g on anduri sisendis pinge 5 mV (valem 3.2), mis on väga lähedal pingenivoole 4,8876 mV. Tulemus teisendatakse digitaalsele kujule ning muundi väljundiks on number üks.

$$U_g = \frac{U_s}{1000} = \frac{5}{1000} = 5 \text{ mV}, \quad (3.2)$$

kus *Ug* on tensoandurile rakendatava koormusele 1 g vastav vaadeldava *ADC* sisendi pinge, mV.

Lähtuvalt arvutustest võib kokkuvõtvalt öelda, et 1 g koormus valitud tensoanduril andis vaadeldava *ADC* väljundisse numbri 1.
Siin aga tuli arvesse võtta mõõtemääramatust ning potentsiaalset müra, mis tähendab, et kalkuleeritud *ADC* väljund oli tegelikult vahemikus 1±1 ühik. Siia lisandus veel korduv mõõtmiste täpsus mis võis mõõtetulemust mõjutada näiteks 5 või 10 ühiku võrra. Lähtuvalt eelnevatest arvutustest tegi töö autor järelduse, et *Atmega32U4* mikroprotsessorisse integreeritud *10 bit* Analoog-digitaalmuunduri kasutamine ei ole selle töö raames sobilik.

Tehes läbi samad arvutused *16 bit ADC-ga* sai autor teada, et 1 g koormust annab *ADC* väljundisse ühiku 65,535 mis ümardatakse 66-ks. See on juba tunduvalt parem tulemus kuid ei pruugi olla piisav.

Tehes arvutused, seekord 24 bitise analoog-digitaalmuunduriga mille väljundiks on number vahemikus 0-16777215 ($2^{24}$- 1= 16777215), sai autor teada, et jagades 5 V vaadeldava muunduri maksimaalse väljundi ühikuga 16777215 on tulemuseks 0,00029802324164052 mV (valem 3.3), mis võrdub muunduri väljundi ühikuga 1. Seega 5 mV sisse mis võrdub 1 g, mahub 16777,215 ühikut (valem 3.4). Töö autor leidis, 24 bitine analoog-digitaalmuundur on sobilik.

$$U_m = \frac{U_s}{16777215} = \frac{5}{16777215} = 0{,}00029802324164052 \text{ mV}, \quad (3.3)$$

$$adcm = \frac{U_s}{U_m} = \frac{5}{U_m} = 16777{,}215, \quad (3.4)$$

kus adcm on ühele grammile vastav vaadeldava *ADC* väljund.

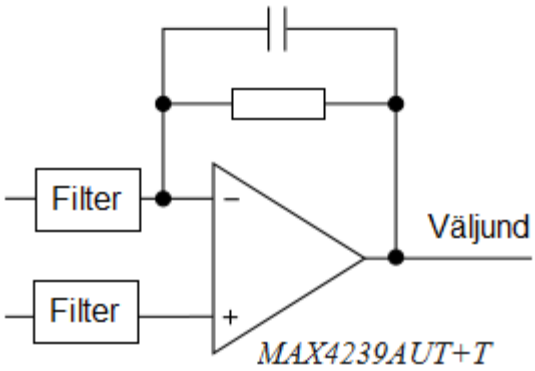
Arvutustest võib kokkuvõtvalt öelda, et 1 g koormus valitud tensoanduril annab vaadeldava *24 bit ADC* väljundisse numbri 16777,215 mis ümardatakse numbriks 16777. Korrutades

16777,215 tuhandega on tulemus 16777215 mis on vaadeldava *ADC* maksimum väljund. See tähendab, et kalkulatsioonid on tehtud õigesti.

*ADC* valik langes integraalskeemi kasutava valmismooduli *HX711* kasuks. Mooduli hinnaks kujunes 1,20 € [14].

## 3.7 Analoogsignaali võimendi valik

Kuna tensoandurist tulev signaal on kahe pinge vahe siis selle suurusjärk jääb ligikaudu 0,2 mV vahemikku. Selline signaal on mikrokontrolleri sisendi jaoks liiga väike ning seda peab võimendama. Võimendiks valis autor üksiktoitega operatsioonivõimendi *MAX4239AUT+T* [15], hinnaga 3,49 €. Komponendi valiku kriteeriumiteks oli pind-montaaži komponent, madal müratase, töö 5 V toitepingega ning sisendsignaali võimalikult lähedane võimendus väljundisse sõltuvalt toitepingest, siit ka suur komponendi omahind. Komponent pidi olema konfigureeritud kui diferentsiaalsignaali võimendi ning olema negatiivselt tagasisidestatud saamaks vajalikku võimendust (joonis 3.3). Võimendus (takisti suurus) tuli katseliselt välja selgitada sõltuvalt filtrite lõppväljunditest, ning tensoandurile rakendatavast jõust, et see sobiks analoog-digitaalmuunduri sisendiks ning ei ületaks võimendi tööpiirkonda kui tensoandurile rakendada maksimaalset koormust milleks on 1 kg.




**Joonis 3.3.** Diferentseeriva võimendi põhimõtteskeem [16]

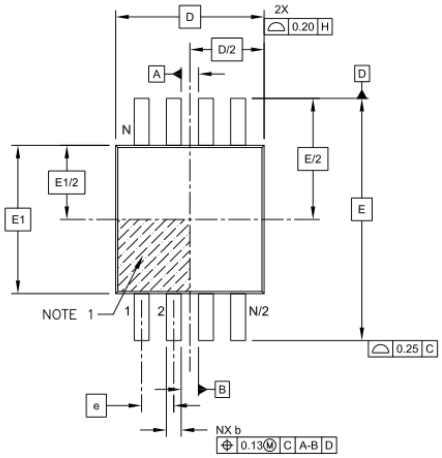
Vastavalt skeemi tööpõhimõttele, operatsioonivõimendi pluss sisendisse rakendatakse pinge, mis on erinev kuid suurem miinus terminali pingest (tensoanduri väljund).

Operatsioonivõimendi hakkab oma väljundis pinget suurendama seni kuni miinus terminali pinge on pluss terminali pingega samastunud teatud vea piires. Siinjuures tekib takistil pingelang. Seega mida suurem on takisti seda suurem on pingelang takistil ning seda suuremaks peab operatsioonivõimendi oma väljundis pinget tõstma, et pingeid pluss ja miinus terminalidel ühtlustada. Tulemusena on võimalik kahe pinge erinevust võimendada. Kondensaator aga aeglustab pinge muutumist, seega vähendades potentsiaalset müra.

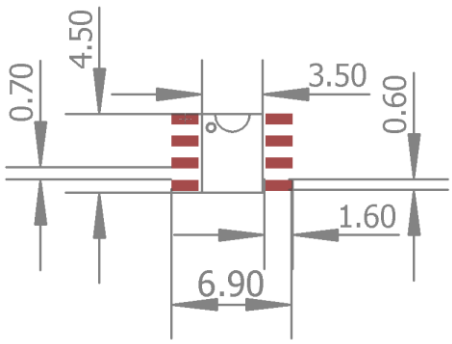
## 3.8 Reaalaja kella valik

Reaalaja kella integraalskeem oli plaanis joota arendusplaadile. Selle tarbeks tuli leida pindmontaaži komponent, mida oleks võimalik mugavalt arendusplaadile joota. Kogemuse põhjal valis töö autor pakendi tüübiks *MSOP* kuna seda on võimalik suure vaevata käsitsi joota. Edasi otsis autor komponenti, mille toitepingeks on võimalik rakendada 5 V, nõutavaks oli ka $I^2C$ liidese ning kellaformaadi *HH:MM:SS* tugi, kus HH tähistab tunde 24 tunni formaadis, MM minuteid ja SS sekundeid. Edasi sai määravaks hind ning andmelehes integraalskeemi reaalseks kasutamiseks piisava info olemasolu (on komponente, mille andmelehes olev info on puudulik). Valik osutus *MICROCHIP MCP79410* kasuks [17], hinnaga 1,84 €.

Vastavalt *MCP79410* andmelehes [18] olevale joonisele (joonis 3.4), kavandas töö autor *Eagle* tarkvaras *MCP79410* komponendi (joonis 3.5).




**Joonis 3.4.** *MSOP* pakendi tehniline joonis (pealtvaade) [18]



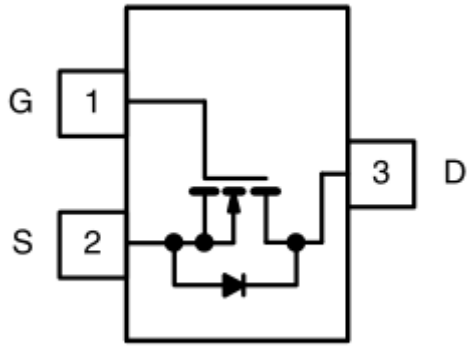

**Joonis 3.5.** *MCP79410* komponendi jootepadjad

Vastav komponendi disain sai kasutusele võetud arendusplaadi disainimisel.

## 3.9 Elektrimootorite ja transistoride valik

Elektrimootoreid valis autor toitepingepinge põhjal, milleks oli 5 V ning teiseks kriteeriumiks pidi tegemist olema reduktormootoriga. See oli oluline vältimaks klapi iseeneslikku avanemist juhul kui mootori võllile rakendatav koormus paneb ta tagurpidi pöörlema. Valitud sai 2 reduktormootorit tööpingega 3-9V [41] .

Kuna kasutada oli 5 V toiteallikas tuli vastavalt sellele valida transistorid (valitud sai *Si2302DDS* [19]). Autor plaanis juhtida mootoreid pingega, selle tarbeks tuli leida kaks n-tüüpi *mosfet* transistori. Kuna mootorid ei pruukinud töö esitamise ajaks Hiinast kohale jõuda või autor mootoritega tegeleda, tuli arvestada variandiga, kus mootorid ei sobi kasutamiseks. Sellisel juhul võis tarvilik olla uute mootorite soetamine mis tarbivad rohkem pinget, näiteks 12V või pinge muundamine suuremaks kui 5 V. Seega transistori üheks oluliseks parameetriks oli vähemalt 12V pinge taluvus transistori paisu ja lätte vahel (joonis 3.6) mis näitab millise maksimaalse pinge korral tootja garanteerib transistori tööd. Ületades etteantud pinge piiri võib transistor küll edasi töötada, kuid tulemus võib olla ettearvamatu.




**Joonis 3.6.** MOSFET transistori sisendid ja väljundid, G – värav, S – läte, D – pais [20].

Järgnevaks kriteeriumiks oli *VGS* pinge värava ja lätte vahel, ehk pinge mis tuleb mikrokontrolleri väljundist, 5 V. Selle pinge juures peab transistor olema täielikult avanenud, s.t et transistori takistus peab olema minimaalne vältimaks komponendi ülekuumenemist.

Viimaseks kriteeriumiks oli maksimaalne voolu läbilaskevõime. Kuna valitud toiteallikas annab välja voolu 2,6 A siis tuli valida vastav komponent mis oluliselt vastavat voolu ei ületaks. Valitud komponent vastas kõigile seatud kriteeriumidele.

## 3.10 Jadaliides

Moodulitevaheliseks andmevahetusliideseks valis autor jadaliidese (*universal asynchronous receiver/transmitter - UART*). Liidesega tööks tuli valida sobiv andmevahetuskiirus, mida kasutada kahe põhimooduli omavahelisel suhtlemisel. Vastavalt *Atmega32U4* andmelehele [13], said valitud järgnevad sätted:

1. *Baud rate*, andmeedastuskiirus 9600 bitti sekundis;
2. *Parity*, paarsus puudub;
3. *Data bits*, andmebittide arv 8;
4. *Stop bits*, stoppbittide arv 1.

Andmeedastuskiirus sai valitud selline, kuna *Atmega32U4* on võimeline töötama kiirusega 16 MHz kuid andmelehe kohaselt [13], sellise taktsagedusega ei ole *UART* suhtlus antud

mikrokontrolleriga võimalik. Seega tuli valida mõni madalam sagedus mida andmeleht kajastas. Töö autor valis mikrokontrolleri töö sageduseks 8 MHz ning andmevahetuskiiruseks 9600 bps (veaprotsendiga 0,2%) ja andmebittide arvuks 8 kuna tegu on siiski 8 bitise mikrokontrolleriga.

## 3.11 Kasutatav tarkvara, veebiraamistikud, programmeerimiskeeled

Autor kasutas antud töö rames järgnevaid programeerimiskeeli ning tarkvara.

Programmeerimiskeeled:

1. *C* – programmeerimiskeel [21];
2. *Python 3.4.6* – programmeerimiskeel [22].

Tarkvara:

1. *Eagle* – elektriskeemide ja trükkplaadi projekteerimise tarkvara 7.7.0, *64 bit* [23];
2. *Atmel Studio 6.0.1996 - Service Pack 2 64 bit* – tarkvarakeskkond arendusplaadi programmeerimiseks, C programmeerimiskeel on integreeritud paketti [24];
3. *Solidworks 2013 Student edition 64 bit* – 3D modelleerimise ja tehniliste jooniste loomise tarkvara [25];
4. *Flip 3.4.7* – Arendusplaadi tarkvaraline programmaator [26];
5. *Hostapd* – lubab luua *hotspot-i*, tarkvara [27];
6. *Iptables – IP* tabelite halduri tarkvarapakett nimega *iptables-persistent;*
7. *Flask* – veebiraamistik, mõeldud veebirakenduse loomiseks [28];
8. *Wtforms* – veebiraamistik raamatukogu *form* objektide haldamiseks *Python-is* [29].

Nii *Flask, Wtforms, Hostapd, iptables-persistent* tuli eraldi *Rasbian* operatsioonisüsteemi installeerida kasutades vastavaid käske:

1. *sudo apt-get install hostapd isc-dhcp-server;*
2. *sudo apt-get install python3-flask;*
3. *pip install WTForms;*
4. *sudo apt-get install iptables-persistent.*

## 3.12 Juhtmete ühendusmoodul

Komponentide jaoks sai disainitud eraldi alus (joonis 3.7) mille külge oli võimalik komponente kinnitada. Samal joonisel on visualiseeritud juhtmete füüsilised ühendused.

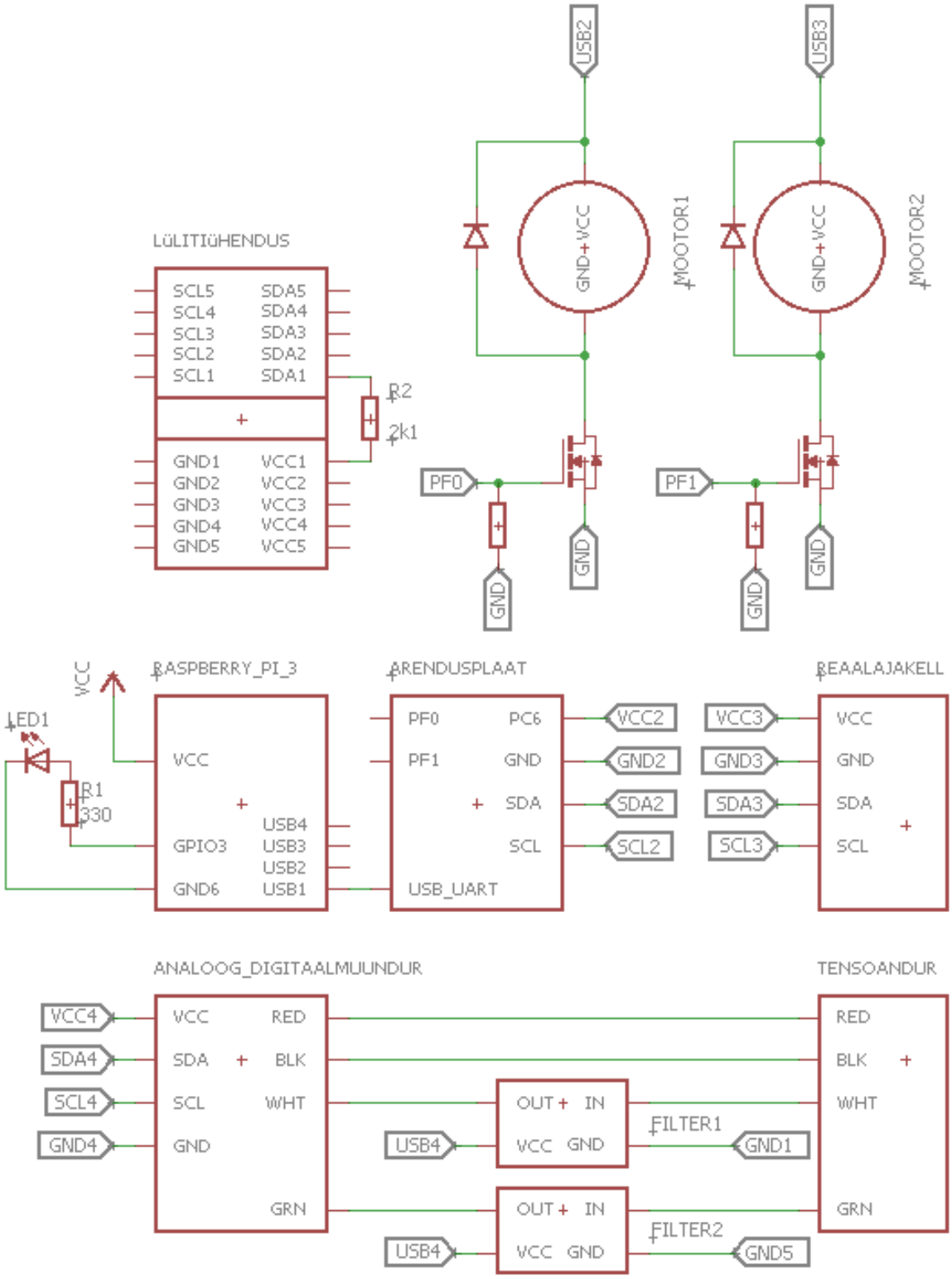
Vooluvõrgust tulev ühendus kinnitub Raspberry Pi toitepistikusse. Kuna *Atmega32U4* mikrokontrolleri arendusplaat sai disainitud nii, et toide võib tulla ka läbi *USB* pistiku, mida kasutatakse *UART* suhtluseks, siis arendusplaadi toide tuleb Raspberry Pi külge ühendatud *USB* kaablist.

Kuna eelnevalt kalkuleeritud voolud jäid alla 20 mA sai disainitud juhtmete ühendusmoodul nimega lülitiühendus, mis koosneb järgnevatest ühendustest (joonis 3.7):

1. 5 V – toitepinge arendusplaadi port 6 väljundviigult;
2. *GND* – maaühendus;
3. *SDA* – jadaliidese andmeliin;
4. *SCL* – jadaliidese takti liin.

Kõik jadaliidese andmeviigud *SDA* ja taktiviigud *SCL* said ühendatud ühendusmooduli vastavate sisendite külge. Sama ühendusmooduli külge said ühendatud arendusplaadi, realalajakella, *ADC* ja tensoanduri maa ühendused ning arendusplaadi, reaalajakella ja *ADC* toiteviigud.

Transistoride kontrollviigud said ühendatud arendusplaadi viikude F0 ja F1 külge ning maaühendused Raspberry Pi *GND* väljaviikude külge. Elektrimootorite toide vastavalt Raspberry Pi *USB* pordist 1 ja 2 kuna iga port suudab välja anda 0,5 A voolu.




**Joonis 3.7.** Juhtmete füüsilised ühendused korpuses

Potensiaalselt võinuks lisada elektrimootorite sisendi ja maaühenduse vahele kondensaatorid et siluda suuri voolumuutusi mootorite käivitushetkel.

## 3.13 Toiteallikas

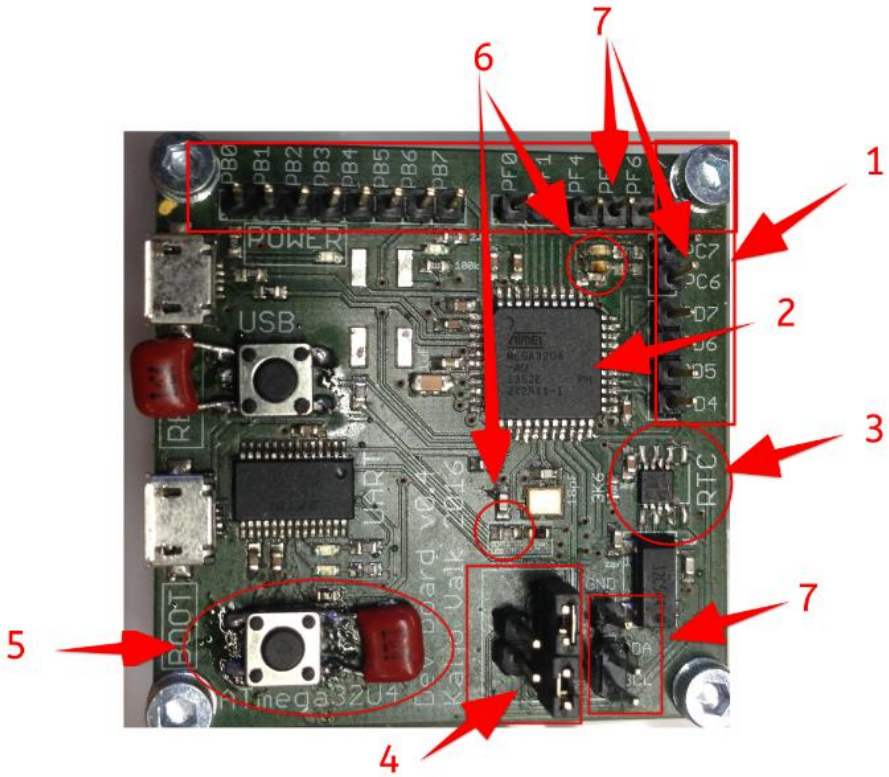
Toiteallikaks valis autor *Raspberry Pi 3 Model B Power Supply 5,1V 2,5A* [30]. Valikus lähtus autor *Raspberry Pi* voolutarbest, mis võib olla halvimal juhul 2,5 A ning *Atmega32U4* voolutarbest, mis on vastavalt andmelehele 8 MHz ja 5 V juures voolutarve 10 mA [13], sinna lisas autor soovituslikud 20 mA iga väljaviigu kohta kuid mitte rohkem kui kõikide viikude pealt kokku kui 100 mA. Reaalaja kella voolutarve patareil ajaarvestusrežiimis maksimaalselt 3 µA ja tüüpiliselt 0,84 µA [18], *HX711 ADC* voolutarve 1,4 mA [31] ja mootorite teoreetiline voolutarve 0,5 + 0,5 A.

# 4. PROTOTÜÜBI VALMISTAMINE

## 4.1 Arendusplaadi valmistamine

Kõigepealt tegi autor valmis arendusplaadi kavandi ning valmis detail sai tellitud Riistvaraprojekti kursuse raames Hiinast. Valmistatud trükkplaat (joonis 4.1) oli rahuldava kvaliteediga. Selle all peab autor silmas fakti kus korduva jootmise käigus üks rada tuli lihtsalt plaadi küljest lahti. See tähendab, et rada oli plaadi küljes halvasti kinni. Võimalik, et tegu oli lihtsalt tootmispraagiga. Rohkem probleeme ega tootjapoolseid vigu autor ei täheldanud.

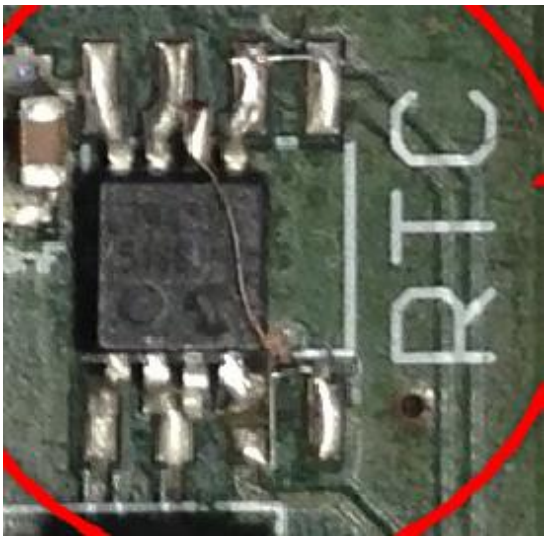
Korduvjootmise käigus said kannatada 3 *LED* indikaatorit, joonisel 4.1 number 6. Samuti enamus lülitiühendusi kuna oli kasutatud plastikust ümbrisega *female pin header* otsikuid, mis tuli asendada *male pin header* komponentidega, mille tulemusena enam korduvjootmisel kahjustusi ei tekkinud, joonisel punkt 7. Lisaks oli autor teinud plaadi markeerimisel vea kus viigud 7 ja 6 olid siiditrükil vahetusse läinud. Probleemi lahendus oli plaadi disaini siiditrükis numbrid ära vahetada. Korduvjootmise käigus said kahjustada ka kõik lülitid, mis tuli samuti asendada, joonisel number 5. Lüliti lülitamisel tekkis digitaalsignaali loogilise nivoo kiire muutus (fenomen nimega *debouncing* – signaali põrkumine). Probleemi oli võimalik lahendada oodates seni kuni signaal stabiliseerub peale lülitust oodates ligikaudu 50 ms. Seda saanuks teha tarkvaraliselt, säästes kulu riistvarakomponentide pealt. Selline lähenemine nõudnuks täiendavat programmeerimist ning ajakulu. Autor leidis, et esialgne otsus probleemi tarkvaraliselt lahendada on liiga ajakulukas ning lisas igale lülitile juurde 100 nF kondensaatori, millega vältis signaali põrkumist riistvaraliselt, joonisel punkt 5.




**Joonis 4.1.** Arendusplaat koos kõigi peale joodetud komponentidega, 1 – mikrokontrolleri väljaviigud P6 ja P7, 2 – *Atmega32U4* mikrokontroller, 3 – *MCP79410* reaalajakell, 4 – jadaliidese lülitusühendused, 5 – lüliti ja 100 nF kondensaator, 6 – *LED* indikaatorid, 7 - mikrokontrolleri väljaviigud

Pöörates tähelepanu kondensaatori ning lüliti ja trükklaadi ühenduskohtadele punktis 5 vaadeldaval joonisel on märgata ebakorrapärast räbusti jäägikihti, mis võib sisaldada näiteks tinatükke sellega luues potentsiaalset lühise võimalikkust. Räbusti jäägid oleks pidanud trükkplaadilt korralikult eemaldama kasutades selleks näiteks propanooli.

Luues juhtmetega ühendusi teiste komponentidega selgus, et arendusplaadil ei ole piisavalt väljaviike komponentide ühendamiseks. Osaliselt nägi autor seda ette ning kavandas jadaliidesele eraldi väljaviigud, kuid 5 V väljundit ette ei nähtud. Seega tuli kasutada mikrokontrolleri väljaviike toite ühenduste loomiseks, siin pidi autor arvestama iga väljaviigu kohta maksimaalsete vooludega ning enne ühenduste loomist testima iga komponendi reaalset voolutarvet.



**Joonis 4.2.** Ebaõnnestunud reaalajakella jootmine, jootepatjade suurus ei lange kokku komponendi suurusega

Omaette probleemiks osutusid reaalajakella jootepadjad, mis said disainitud vale komponendi jaoks (joonis 4.2). Asenduseks sai valitud reaalajakella moodul *DS3231* [40], mis töötas akupatareil mis on kallis ning seda ei pruugi olla võimalik hiljem saada. Autor uuris Eesti poodides saadaolevat patareide valikut ning leidis, et akupatarei on võimalik asendada tavalise 3 V liitiumpatareiga, näiteks *CR2032* mahtuvusega 210 mAh [32] (selleks tuli moodulil lahti ühendada akupatareid toitev ühendus eemaldades selleks vastav takisti, mis sai hiljem ka tehtud). Veendumaks et patarei on suuteline piisava aja vältel kella vooluga varustama, on tehtud kalkulatsioon (valem 4.1) toiteallika kestvusele.

Vastavalt reaalajakella *DS3231* andmelehele [33] on komponendi voolutarve ajalugemisel maksimaalselt 3 µA ja vooluleke välise lisatoiteallika korral maksimaalselt 100 nA.

$$t = \frac{Pm}{I} = \frac{210}{0{,}0031} = 67741{,}935 \text{ h}, \tag{4.1}$$

kus t on aeg tundides mille jooksul on patarei suuteline välja andma 3.1µA voolu pingel 3 V, h;

$Pm$ – patarei mahtuvus milliamprit tunnis, mAh ;

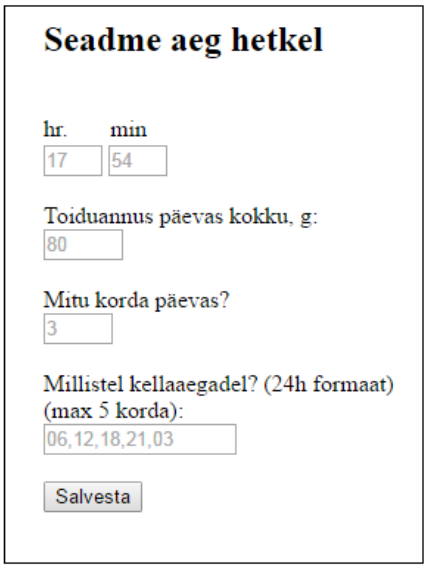
I – kogu kella voolutarve, mA.

Arvutustest selgus, et antud patarei on suuteline toitma moodulit 67741,935 tundi (7,733 aastat) mis oli igati sobilik valik. Tuleb märkida, et arvutatud sai teoreetiline suurus. Reaalses elus on see number mõnevõrra väiksem.

Reaalajakella ja *Atmega32U4* arendusplaadi omavaheliseks suhtluseks tuli autoril kirjutada vastav tarkvaraline kood mis võimaldas kellamooduli ja arendusplaadivahelist suhtlust (LISA K).

## 4.2 Seadmete programmeerimine

Kommunikatsiooni, veebiserveri ja veebilehe tarbeks lõi autor vastava koodi (LISA L), mille arendusplaadi osapoole eesmärk on oodata sissetulevaid andmeid läbi *UART* liidese kasutades programmilisi katkestusi, salvestada vastavad sätted mikrokontrolleri püsimällu (et nad ei kaoks voolukatkestuse tagajärjel) ning siis saata saadud sätted tagasi läbi *UART* liidese *Raspberry Pi-le*.




**Joonis 4.3.** Veebilehel kuvatav kasutajaliides sätete seadistamiseks

*Raspberry Pi* koodi kirjutamisel lähtus autor objektorienteeritud koodi kirjutamise võimalikkusest *Python-is* kus lõi klassi *Settings*, vastavad konstruktorite, Getterite ja Setterite meetodid, mida kasutas klassisiseste muutujate (sätete) haldamiseks. Koodi

ülesandeks oli saata arendusplaadile sätteid, neid vastu võtta ning kuvada veebilehel. Lisaks kuvada veebilehte järgneval veebiaadressil*: http://192.168.42.1:8080* (joonis 4.3) kus on kuvatud lahtrid sätete sisestamiseks. Kasutajale on ette antud näide õrnas kirjas, kuidas lahtreid täita, vältimaks situatsiooni kus kasutaja ei tea mida sisestada. Lisaks sellele kuvatakse kasutajale võimalikult vähe informatsiooni, et kasutajaliides ebavajalikust infost puhtana hoida, samas piisavalt oma eesmärgi täitmiseks lähtuvalt (ptk 2.5) püstitatud kasutajaliidese mittefunktsionaalsetele nõuetele.

## 4.3 Kasutajaliidese seadistamine

Veebilehe kuvamiseks tuli autoril installeerida vastav tarkvara (peatükk 3.10) ja see konfigureerida. Eesmärk oli luua lokaalne *Hotspot* nimega *CatSpot*, turvata *WPA2* parooliga ning lubada kasutajal sellega ühendust luua sisestades brauserisse vastav *IP* aadress. Selleks tuli *Raspberry Pi-le* installeeritud operatsioonisüsteemi kõvakettal modifitseerida */etc/dhcp/dhcpd.conf* faili järgnevalt:

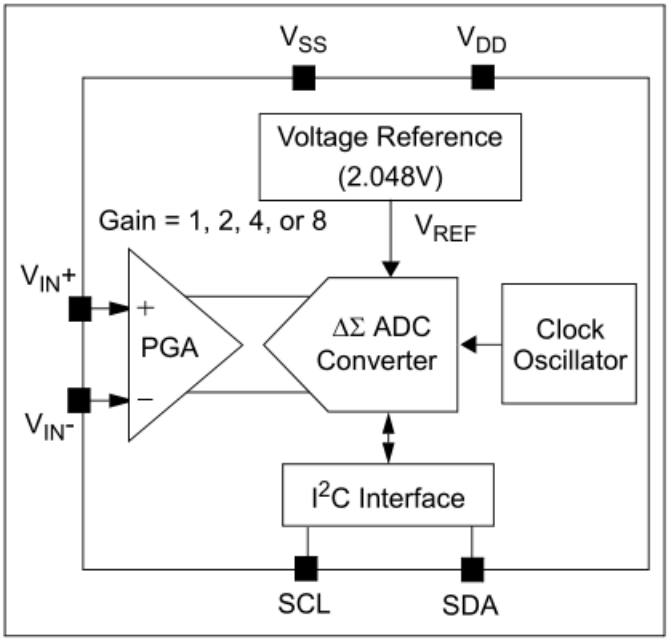
```
interface=wlan0

ssid=CatSpot
country_code=EE

# Kasuta 2.4GHz
hw_mode=g

# Kasutuses olev kanal
channel=6

# Luba kõiki MAC aadresse
macaddr_acl=0
auth_algs=1

# Kasutaja peab teadme võrgu nime ühendamiseks
ignore_broadcast_ssid=0

# Kasuta WPA2-e
wpa=2

wpa_passphrase=12345678
wpa_key_mgmt=WPA-PSK

# KAsuta AES-t mitte TKIP
wpa_pairwise=CCMP

wpa_group_rekey=86400

# Aktiveeri 802.11n
ieee80211n=1
wme_enabled=1
```

**Joonis 4.4.** *Hotspoti* seadistamine, */etc/dhcp/dhcpd.conf* faili sisu

Lisaks andis autor *Raspberry Pi-le* staatilise *IP* aadressi, milleks valis *http://192.168.42.1:8080*. Selleks modifitseeris */etc/network/interfaces* faili kuhu lisas valitud *IP*.

## 4.4 Tensoandurilt andmete lugemine

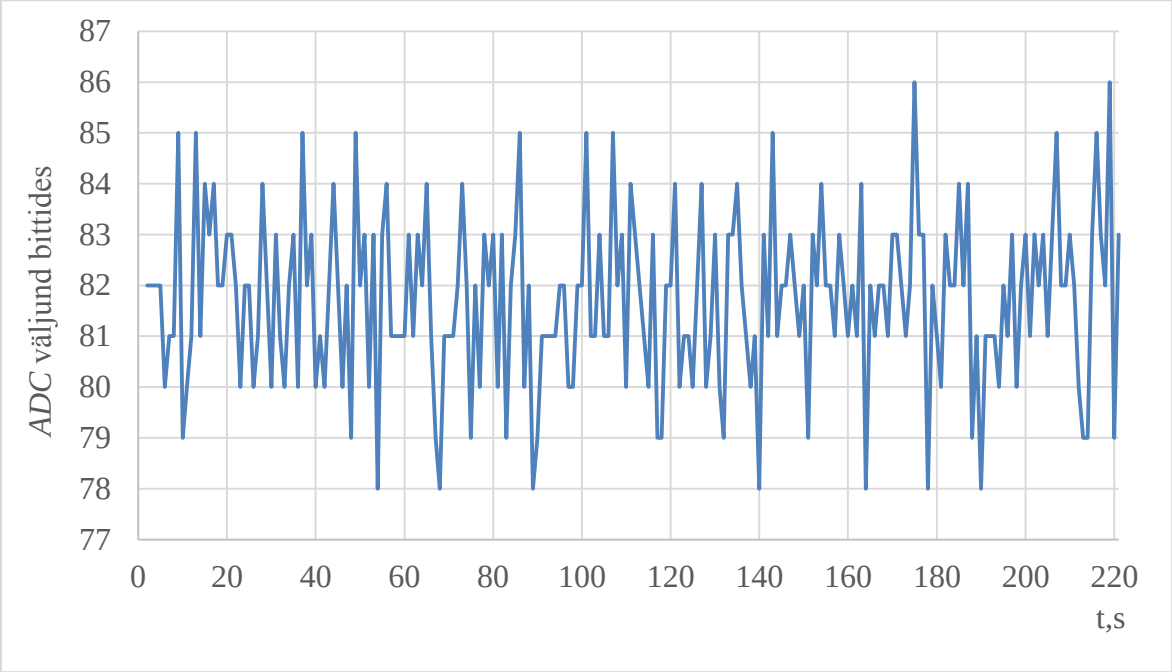
Võttes käiku valitud *ADC* valmismooduli *HX711* [14] püüdis autor luua komponendi jaoks koodi kasutades internetis olevaid näiteid erinevate mikrokontrollerite jaoks kuid komponendist tulev *ADC* väljund oli vahemikus 3-200 ühikult. Komponendiga kaasas olev dokumentatsioon oli aga puudulik mis ei võimaldanud selgusele jõuda miks seade nii käitub. Valmismoodulist tuli loobuda ja valida uus *ADC*. Valik langes *16 bit ADC MCP3425* [34] integraalskeemi kasuks. Uurides komponendi plokkskeemi tundus valik sobilik (joonis 4.5). Diferentseeruv signaali sisend, jadaliidese tugi, *16 bit* resolutsioon, *SOT-23* pakend (mugav käsitsi joota), töö toitepinge vahemikus 2,7 V-5,5 V.




**Joonis 4.5.** *MCP3425, 16 bit ADC* plokkskeem [35]

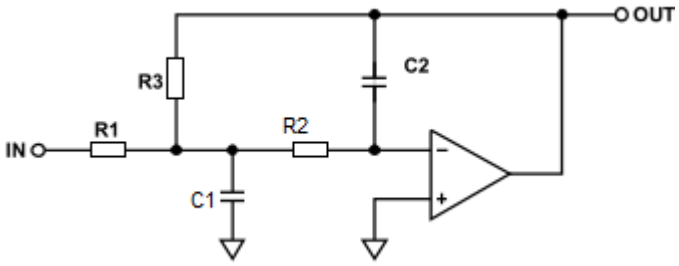
Kuna komponent kasutab jadaliidest, tuli autoril kirjutada vastav tarkvaraline kood võimaldamaks valitud *ADC* suhtlust läbi jadaliidese (LISA M).

Analoogisignaali mõõtmiseks ühendas autor *ADC* sisendi tensoandurist tuleva võimendatud signaali külge, kusjuures *ADC* üks sisend tuli ühendada maaühenduse külge, tegu oli diferentsiaal *ADC-ga*. Võimendil oli aga ainult üks väljund. (võimalik et oleks pidanud valima ühe sisendiga *ADC*, vältimaks maaühendusest potentsiaalset tulevat müra). Autor mõõtis *ADC* väljundit. Tehtud mõõtetulemustest selgus, et ümbritsev müra oli signaali kätte saamiseks liiga suur, näidatud joonisel 4.6 kus ordinaatteljel on mõõdetud 16 *bit ADC* väljund bittides ning abstsiss teljel on aeg sekundites. Mõõtmisi sai korratud 3 korda ning tulemus oli joonisel näidatule ligilähedane, kusjuures iga mõõtetulemuse korral sai rakendatud füüsilist jõudu tensoandurile, asetades andurile kolmeks sekundiks perioodiliselt 10 g massiga eset.




**Joonis 4.6.** *16 bit ADC* mõõtetulemus

Müra summutamiseks tuli autoril teha 2 madalsagedusfiltrit (LISA C) ning paigaldada nad enne signaali võimendamist (joonis 4.8), vältimaks müra võimendamist. Selleks arvutas ta kõigepealt takistite ning kondensaatorite väärtused (valem 4.2) valides sobivad, et summutatav sagedus jääks alla 20 Hz. Siinjuures R2, R3, C1, C2 vastavad takistite ja kondensaatorite paiknemisele joonisel 4.7 ning summutatav sagedus valemis 4.2 on avaldatud [36].




**Joonis 4.7.** Mitmelt tagasisidestatud madalsagedusfilter [37]

$$fc = \frac{1}{2\pi\sqrt{R2 \cdot R3 \cdot C1 \cdot C2}} = \frac{1}{2\pi\sqrt{5000 \cdot 5000 \cdot 4{,}7 \cdot 10^{-6} \cdot 1 \cdot 10^{-6}}} \approx 14{,}683 \text{ Hz}, \quad (4.2)$$

kus *fc* on minimaalne summutatav sagedus, Hz;

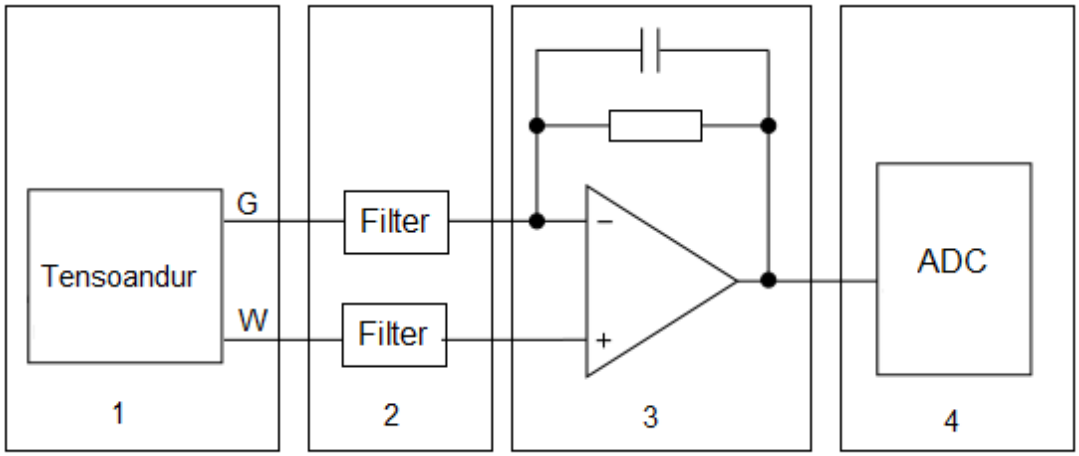
*R2* – 5 kΩ takisti, Ω;

*R3* – 5 kΩ takisti, Ω;

*C1* – 4,7 µF kondensaator, F;

*C2* – 1 µF kondensaator, F;

Kalkulatsioonist lähtuvalt kõik signaalid sagedusega üle 14,683 Hz summutatakse.

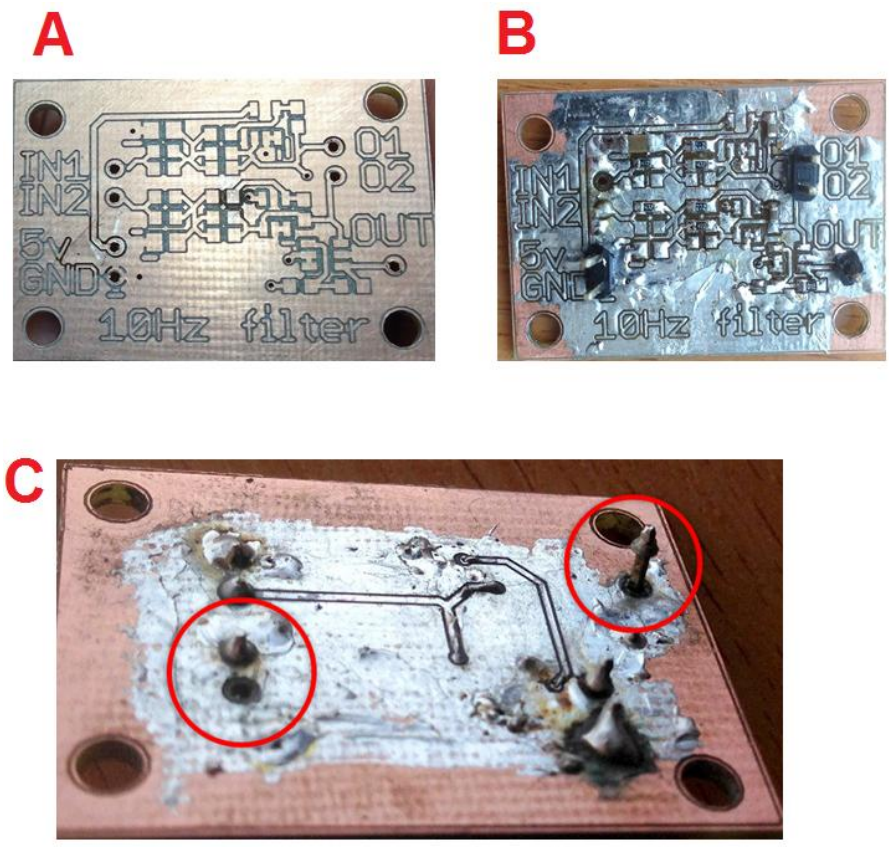



**Joonis 4.8.** Filtrite paiknemine kus 1 – tensoandur, 2 – filtrid, 3 – negatiivselt tagasisidestatud operatsioonivõimendi, 3 – *ADC*

Filtrite trükkplaat sai freesitud. Tulemus on näidatud joonisel 4.8 osa A. Autor teadis, et vask oksüdeerub ning jätab koleda pinna, seega otsustas katta plaadi üleni tinaga. Arvestades trükkplaadi mõõtmeid ei osutunud see raskeks. Tulemus aga ei olnud see mida autor ootas.

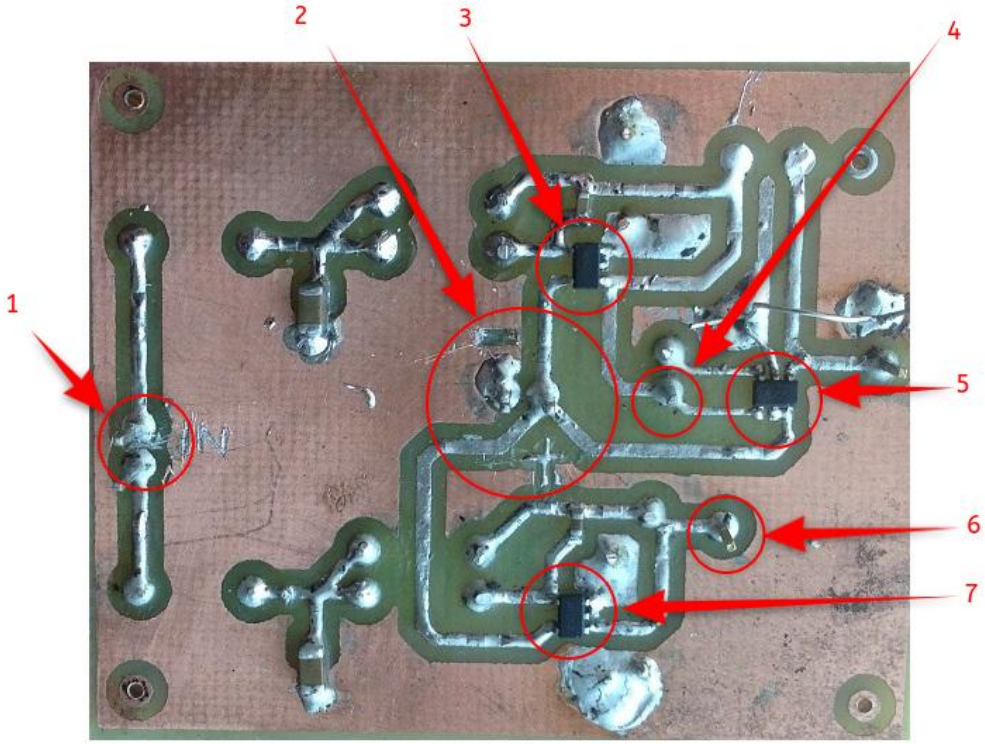
Esteetiliselt jäi trükkplaadi pind siiski koledaks (joonis 4.9 osa B). Freesitud trükkplaadil oli ka üks tõsine defekt. Nimelt trükkplaadi läbiava ühendused ei suutnud pistikühendusi kinni hoida ning viimased murdusid trükkplaadi küljest ära (joonis 4.9 osa C). Läbiavaühendused said tehtud identselt arendusplaadi läbiavadega kuid viimasel taolist probleemi polnud. Probleemi oleks võinud lahendada suurendades kinnihoidvat vase läbiava pinda.

Autoril õnnestus siiski testimiseks vajalikud ühendused luua, kuid filtrid nii nagu oli oodatud tööle ei hakanud. Arvestades trükkplaadi mõõtmeid ei hakanud autor viga otsima vaid kavandas uue, suurema 45 x 55 mm trükkplaadi (joonis 4.10) ning seekord söövitas selle. Kogu trükkplaadi valmistamise protsess oli seekord tunduvalt kiirem kuid mitte niivõrd täpne kui freesides. Lisaks oli trükkplaati disainides autor kiirustanud ning ülemisele peakihile jäid märkimata ühendusi tähistavad kirjed (autor lõikas need hiljem juurde käsitsi kasutades mikroskoopi ja skalpelli). Trükkplaadi mõõtmed olid suuremad ning see võimaldas kergemini viga otsida.




**Joonis 4.9.** Filtrite trükkplaat. A – freesitud trükkplaat enne jootmist, B – freesitud trükkplaat kaetud tinakohiga, C – freesitud trükkplaat (vead).

Kontrollides trükkplaati, eemaldades ühe lühise ning pingestades, selgus, et üks filter tarbis 20 mA voolu, mõlemad kokku 40 mA. Signaalivõimendi aga üle 100 mA. Disainitud oli aga toide ühe mikrokontrolleri väljaviigu pealt milleks oli 20 mA. Autor testis filtreid autonoomse vooluallika pealt, vältimaks potentsiaalset kahju arendusplaadile järgmisena ühendas võimendi lahti ning testis filtreid eraldi. Filtreeritud signaal oli tunduvalt stabiilsem võrreldes filtreerimata signaaliga. Mõõtes mõlema filtri väljundi pingenivoo erinevust sai autor tulemuseks 90 mA mis oli lubatud tulemus, paraku signaal ei muutunud hetkel kui tensoandurile rakendati füüsilist koormust. Ilma filtriteta aga andsid tensoanduri väljundid pingete erinevuseks 2 mV mis muutus kuni 20 mV ajal kui tensoandureile rakendati mõõdukat füüsilist koormust.




**Joonis 4.10.** Söövitatud filtrite trükkplaat, kus 1 – filtrite sisendid, 2 – toide + ja mandus -, 3 – esimese filtri operatsioonivõimendi, 4 – esimese filtri väljund, 5 – mõlema filtri diferentseeruva signaali operatsioonivõimendi, 6 – teise filtri väljund, 7 – teise filtri operatsioonivõimendi

Võimalik olnuks veel testida ühe filtri ning ilma filtrita signaalide käitumist kuid selle töö raames tuli ajapiirand peale ning sellega filtrite katsed piirdusid ja müra probleemi lahendus jäi tulevikku.

## 4.5 Komponentide nimekiri ja hinnad

Seadme ligikaudne komponentide omahind on näidatud tabelis (tabel 3.2). Kuna arendusplaat koosneb paljudest komponentidest mis said tellitud veebipoest [38] siis ei hakanud autor tabelisse eraldi komponente välja tooma vaid märkis ära arendusplaadi kogumaksumuse. Arendusplaadi komponentide nimekiri on lisatud lisana B. Vineeri maksumuse uuris autor minnes kohale ja küsides hinnapakkumist firmast *ECCOM*.

**Tabel 3.2.** Komponentide hinnad

| | |
|---|---|
| *Raspberry Pi 3 model B* | 59 € [39] |
| Arendusplaat | 25 € |
| *DS3231*, reaalajakell | 1,29 € [40] |
| *MAX4239AUT+T*, operatsioonivõimendi | 3,49 € [15] |
| *MICROCHIP MCP79410, reaalajakell* | 1,84 € [17] |
| *HX711, ADC* | 1,20 € [14] |
| *MCP3425, 16 bit ADC* | 2,2 € [34] |
| Vineer | 30 € |
| Tensoandur | 2,31 € [12] |
| Elektrimootorid | 5 € [41] |
| *SI2302DDS* transistorid 2x | 1 € [19] |
| Hind koku | 132 € |

Seadme omahind on seega 132 €.

# KOKKUVÕTE

Antud töö eesmärk oli projekteerida automaatne kuiva kassitoidu jaotur mille seadistamine toimuks läbi veebiliidese kohtvõrgus ja ei vaja selleks internetiühendust. Lisaks saab seadet konfigureerida hoolimata kasutatavast brauserist.

Peatükis 1 uuris autor kassitoiduga seonduvat teoreetilist baasi, sealhulgas kuivtoidu maksumust, tootjaid ja Eestis olevat kassitoidu turgu. Leidis optimaalse toidupakendi ja selle hinna lähtuvalt toidu säilitusajast. Uuris kuivtoidu säilitamise ja hügieeninõudeid, kasside toitmise metoodikat ning leidis kuni kuue kiloste kassidele soovituslikud päevased toiduannused, millest lähtus hiljem kaalukaussi projekteerides. Peatüki lõpus tegi autor turuuuringu, milles tegi ülevaate kaheksast olemasolevast kassitoidu jaoturist ning tõi välja seadmete olulisi puuduseid.

Peatükis 2 püstitas autor sissejuhatuses määratletud ülesande täitmiseks lähteülesande ja määratles probleemi, mida lahendama hakkas. Selleks analüüsis kes, miks, kus, millal ja kuidas vaadeldavat seadet kasutama hakkab. Enne nõuete püstitamist pani autor paika elementaarsed nõuded nõuetele ning arutles selle üle mis saab kui hilisemas projekti staadiumis osutub vajalikuks püstitatud nõudeid muuta, eemaldada või lisada. Järgnevalt visualiseeris projekteeritava seadme arhitektuuri skeemi näol, millel näitas moodulite omavahelist kommunikatsiooni ja sellega kogu seadme kontsepti. Pani paika funktsionaalsed ja mittefunktsionaalsed nõuded.

Peatükis 3 kavandas autor arendusplaadi trükkplaadi, selgitas millistest kriteeriumidest lähtus antud projekti raames ning pani paika arendusplaadi tööülesanded. Järgnevalt projekteeris seadme korpuse ning sisedetailid. Selgitas millist tarkvara tuleb kasutada kasutajaliidese loomisel. Järgnevalt visualiseeris arendusplaadi ja Raspberry Pi mooduli vahelise kommunikatsiooni. Selgitas milliste kriteeriumide alusel valis tensoanduri. Kalkuleeris vajaliku resolutsiooni mille alusel valis sobiliku analoog-digitaalmuunduri. Põhjendas analoogsignaali võimandi, reaalajakella, elektrimootorite ja transistoride valikut.

Selgitas millise liidese valis mooduli tevaheliseks suhtluseks ning valis suhtluseks vajalikud parameetrid. Tõi välja nimekirja projekti raames kasutatavatest programmeerimiskeeltest ning tarkvarast. Disainis juhtmete füüsiliste ühenduste ühendusmooduli korpuses, visualiseeris ühendused vastaval joonisel ning selgitas lahti ühenduste põhimõtte. Peatüki lõpus selgitas toiteallika valikut.

Peatükis 4 tõstis esile valminud arendusplaadi. Tõi välja trükkplaadi ja jootmise käigus esinenud probleemide lahendusi. Samuti valis uue reaalajakella tehes vajalikud kalkulatsioonid, muudatused mooduli füüsilisse struktuuri ning kirjutades valmis vastava tarkvaralise koodi reaalajakella ja arendusplaadi vahelise suhtluse tarbeks. Seadistas *Raspberry Pi* edastamaks parooliga kaitstut kohtvõrguühendust, programmeeris veebiserveri, veebilehel kuvatava kasutajaliidese ning arendusplaadi ja *Raspberry Pi* vahelise kommunikatsiooni. Selgitas uue *ADC* valikut ning katsetas vaadeldavat komponenti luues vajaliku tarkvaralise koodi *ADC* ja arendusplaadi vaheliseks suhtluseks kasutades $I^2C$ liidest. Tõi välja *ADC* katsetulemused ning lõi kontsepti analoogsignaalis oleva liigmüra probleemi lahendamiseks. Selleks kalkuleeris madalsagedusfiltri poolt summutatava sageduse ning kavandas kaks trükkplaati vajalike filtritega. Selgitas freesitud ja söövitatud trükkplaadiga esinenud probleeme ning pakkus välja lahendusi. Peatüki lõpus tõi autor välja oluliste komponentide nimistu ning seadme omamaksumuse.

Seade sai vastavalt sissejuhatuses püstitatud eesmärkidele valmis projekteeritud. Valmistati ka osaline seadme prototüüp ning testiti elektroonsete süsteemide toimimist. Valmis jaoturi tarkvaraline juhtimisloogika. Projekteeritud seadme kasutajaliides ja kasutajavaheline interaktsioon on veebipõhine. Vastavalt moodulite funktsioonile toimib komponentide-vaheline andmeside. Valmis seadme modulaarne baas mis võimaldab tulevikus väikse ajakuluga projekti uute komponentide integreerimist. Seadet on võimlik edasi arendada lisades klapi, mis seadme lõplikult hermetiseerib, ventilaatori mis toidukonteinerist osaliselt õhu välja pumpab ja avaga küljepaneeli koos vajaliku kaalumiskausi sulguriga.

# SUMMARY

The aim of this thesis was to design an automatic dry cat food dispenser with the possibility of configuring the device entirely through a local web interface, without requiring an internet connection and to be independent of widely used operational systems.

In chapter 1 the author researched the market and the cost of the cat food in Estonia. Found the optimal food packaging and pricing based on the package size, price and the food shelf life offered by the cat food manufacturers. Examined ways to store dry food based on the hygiene requirements. Researched up to six kilos feline feeding methods and found recommended daily feeding quantities which he later used to design the weighing bowl. At the end of the chapter, the author gave an overview of the eight existing cat food dispensers, and highlighted the substantial deficiencyes of the equipment.

In chapter 2 the author set the initial task and defined the problem, which he began to settle. For this purpose he analyzed who, why, where, when and how the device will be used. Before erecting the requirements the author decided to set up elementary requirements for the requirements and discussed what happens if during the project at a later stage, it becomes necessary to set up new, change, add or remove requirements. Next, visualized the architecture of the projected device via a diagram form, which showed the entire concept of the device and communication between the modules. Set up functional and non-functional requirements.

In chapter 3, the author designed the development circuit board explained the criteria used for the design as well as put in place the tasks of the development board. Subsequently, the device body and the internal parts of the housing were designed. Also explained what software should be used when creating the user interface. Next, a visual image of the inter-module communication between Raspberry Pi and a development circuit board was created. Explained what the criteria the strain gauge was chosen by. Calculated the required resolution that was used to choose a suitable ADC. Explained the choice reasons for an

analog signal amplifier, real-time clock, electric motors and transistors. Explained which communication interface was chosen and selected the required parameters for inter-module communication. Created a list of a software, frameworks and programming languages, which are to be used within the project. Designed the physical wire connections that are to be used to connect to the module housing, visualized the connections and explained the principle of choices. At the end of the chapter, explained the choice of the power supply.

In chapter 4, the author showed the completed development board. Pointed out problems with soldering of the board and offered solutions. A new real time clock had to be chosen, for this purpose the author performed the necessary calculations, made changes in the physical structure of the module itself and wrote a software code to enable communication between the development board and the clock. Configured Raspberry Pi to transmit a password-protected wireless local area network connection programmed a web server, a website and a communication routine to enable communication between Raspberry Pi and the development circuit board. Explained the selection of a new ADC, tested the components by creating the necessary software code to enable the ADC and the development board to interact using the $I^2C$ interface. Displayed the ADC test results and tried to solve the analog noise problem by calculating the analog signal cutoff frequency of the multiple feedback low pass filter and designing two filter circuit boards. Explained problems encountered with machined and etched circuit board and suggested solutions. At the end of the chapter, there is a list of the most important components and the total price of the device.

The device was designed in according to the set goals in the introduction. A partial prototype was built and a test of electronic system was performed to ensure the inter-module communication was working. Device software control logic was completed. Design, programming of the device user interface and web-based user interaction was made. According to the function of the modules the communication between modules is working. A modular base was created to enable future integration of new components into the project with less effort and time. The device can be further developed by adding a valve to seal the device permanently, a fan can be attached to the food container to pump the air out of the food container. A side panel with opening and necessary closure element for the weighing bowl.

# KIRJANDUSE LOETELU


**1.** Worldwide Pet Ownership Statistics. (2017). *PetSecure***.**
http://www.petsecure.com.au/pet-care/a-guide-to-worldwide-pet-ownership/ (24.02.2017).

**2.** Kassitoit, kassitoidud, royal canin kassitoit advance kassitoit orjen. *Gardening OÜ.*
http://kodukaubamaja.ee/lemmikloomatarbed-kassitoidud/#/ (28.02.2017).

**3.** Вопросы и ответы. *Royal Canin.*
http://www.royalcanin.com.ua/royal-canin2/nashi-kontakty/voprosy-i-otvety (02.03.2017).

**4. Davidson, C.** (2012). Kassipidaja käsiraamat: Tallinn: Varrak, lk 102.

**5.** Obesity Facts and Risks. (2017). *Association for Pet Obesity Prevention.*
http://petobesityprevention.org/ (02.03.2017).

**6.** Советы экспертов по выбору правильного корма и диеты на официальном сайте Royal Canin, Чем правильно кормить стерилизованную. *Royal Canin.*
www.royal-canin.ru/cats/pravilnoe-pitanie/chem-pravilno-kormit-kastrirovannogo-kota/
(11.03.2017).

**7. Ávila, L.** (2017). Best 15 Automatic Cat Feeder Comparison Chart. *Cat food dispender reviews.*
https://www.catfooddispensersreviews.com/best-15-automatic-cat-feeder-comparison-chart/ (10.03.2017).

**8. Qureshi, A., Memon, M. Z., Vazquez, G,. Fareed, M,. Suri, K.,** (2009). Cat ownership and the Risk of Fatal Cardiovascular Diseases. Results from the Second National Health and Nutrition Examination Study Mortality Follow-up Study. *–Journal of Vascular and Interventional Neurology*. Jan; 2(1): 132–135.

**9.** Tervisest tingitud igapäevategevuste piiratus soo ja vanuserühma järgi. T*ervisestatistika ja terviseuuringute andmebaas.*
http://pxweb.tai.ee/PXWeb2015/sq/b9e7651e-4604-4c92-87d3-5c00c85f6aa4
(10.04.2017).

**10. Jensen, J., J.** (2012). User Interface (UX) Techniques. *Trifork.*
http://www.e-fork.de/news/2014/9/12/9whlpp5ui9vr9nwu81o3ng02qu7tn6 (10.04.2017).

**11.** Setting up a Raspberry Pi as a WiFi access point. *Adafruit Learning System.* https://learn.adafruit.com/setting-up-a-raspberry-pi-as-a-wifi-access-point/install-software (10.04.2017).

**12.** Digital Electronic Scale 1Kg Weight Weighing Sensor Load Cell DC 3V 12V-in Sensors from Electronic Components. *Alibaba Group.* https://www.aliexpress.com/item/Digital-Electronic-Scale-1Kg-Weight-Weighing-Sensor-Load-Cell-DC-3V-12V/32637105903.html?spm=2114.13010608.0.0.IAD4K9 (11.05.2017).

**13.** ATmega16U4/32U4, 8-bit Microcontroller with 16/32K bytes of ISP Flash and USB Controller. *Atmel*. http://www.atmel.com/Images/Atmel-7766-8-bit-AVR-ATmega16U4-32U4_Datasheet.pdf (11.05.2017).

**14.** HX711 high-precision electronic weighing sensor 24 bit A / D converter board. *Alibaba Group.* https://www.aliexpress.com/item/HX711-high-precision-electronic-weighing-sensor-24-bit-A-D-converter-board/32417455391.html?spm=2114.13010308.0.0.aTdSmc (11.05.2017).

**15.** MAX4239AUT+T - MAXIM INTEGRATED PRODUCTS - Operational Amplifier, Precision, 1 Amplifier, 6.5 MHz, 1.6 V/µs, 2.7V to 5.5V, SOT-23, 6 Pins. *Farnell element14 Eesti.* http://ee.farnell.com/maxim-integrated-products/max4239aut-t/op-amp-6-5mhz-1-6v-us-sot23-6/dp/2511028 (11.05.2017).

**16.** Ultra-Low Offset/Drift, Low-Noise, Precision SOT23 Amplifiers. *Maxim Integrated*. http://www.farnell.com/datasheets/2001029.pdf?_ga=2.234664734.1036087867.1495544668-678882859.1495517106 (11.05.2017).

**17.** MCP79410-I/MS - MICROCHIP - RTC IC, Date Time Format (Date/Month/Year hh:mm:ss), I2C, Serial, 1.8 V to 5.5 V, MSOP-8. *Farnell element14 Eesti.* http://ee.farnell.com/microchip/mcp79410-i-ms/rtcc-12c-1k-ee-64b-sram-8msop/dp/1823153?st=MCP79410&pf=111868735&clock-ic-case-style=msop&anyFilterApplied=true&ddkey=http%3Aet-EE%2FElement14_Estonia%2Fw%2Fc%2Fsemiconductors-ics%2Fclock-timing-frequency-management%2Freal-time-clocks (11.05.2017).

**18.** Battery-Backed I2 C Real-Time Clock/Calendar with SRAM, EEPROM and Protected EEPROM. *Microchip Technology Inc.* http://www.farnell.com/datasheets/2244078.pdf?_ga=2.214066455.1063931271.1494521390-226024234.1479828035 (11.05.2017).

**19.** SI2302DDS-T1-GE3 - VISHAY - MOSFET Transistor, N Channel, 2.6 A, 20 V, 0.045 ohm, 4.5 V, *Farnell element14 Eesti.* http://ee.farnell.com/vishay/si2302dds-t1-ge3/mosfet-n-ch-20v-2-6a-sot-23-3/dp/2400359?ost=Si2302DDS&anyFilterApplied=false&searchView=table&iscrfnonsku=false&ddkey=http%3Aet-EE%2FElement14_Estonia%2Fsearch (11.05.2017).

**20.** N-Channel 20 V (D-S) MOSFET, *Vishay.* http://www.farnell.com/datasheets/2049391.pdf?_ga=2.10975991.213492035.1495451831-226024234.1479828035 (10.05.2017).

**21.** Learn C and C++ Programming, *C programming.com.* http://www.cprogramming.com/ (10.05.2017).

**22.** Python.org, https://www.python.org/ (10.05.2017).

**23.** EAGLE, PCB Design Schematic Software, *Autodesk.* https://www.autodesk.com/products/eagle/overview (10.05.2017).

**24.** Atmel Studio, www.atmel.com/tools/atmelstudio.aspx (10.05.2017).

**25.** SOLIDWORKS Student Edition Software, http://www.solidworks.com/sw/industries/student-edition.htm (10.05.2017).

**26.** Flexible In-system Programmer, *Atmel.* http://www.atmel.com/tools/flip.aspx (11.05.2017).

**27.** Hostapd: IEEE 802.11 AP, IEEE 802.1X/WPA/WPA2/EAP/RADIUS Authenticator, http://drvbp1.linux-foundation.org/~mcgrof/rel-html/hostapd/ (11.05.2017).

**28.** Flask (A Python Microframework), http://flask.pocoo.org/ (11.05.2017).

**29.** WTForms Documentation, http://wtforms.readthedocs.io/en/latest/# (11.05.2017).

**30.** Alibaba Group, aliexpress.com, Official Raspberry Pi 3 Model B Power Supply 5,1V 2,5A, https://www.aliexpress.com/store/product/New-Official-Raspberry-Pi-3-Model-B-Power-Supply-5-1V-2-5A-Micro-USB-Power/615778_32691589127.html?src=google&albch=search&acnt=479-062-3723&isdl=y&aff_short_key=UneMJZVf&albcp=266121556&albag=7593673036&slnk=&trgt=dsa-125560162636&plac=&crea=64152518596&netw=g&device=c&mtctp=b&memo1=1t2&gclid=CNuRl8W0-dMCFREdGAodRtUMvw (11.05.2017).

**31 .** *HX711* 24-Bit Analog-to-Digital Converter (ADC) for Weigh Scales. *Avia Semiconductor.* https://cdn.sparkfun.com/datasheets/Sensors/ForceFlex/hx711_english.pdf (11.05.2017).

**32.** CR2032 Lithium Battery Coin Cell. M*ulticomp*. http://www.farnell.com/datasheets/1671733.pdf?_ga=2.243386114.628336851.1495370223-226024234.1479828035 (11.05.2017).

**33.** DS3231 Extremely Accurate I2C-Integrated RTC/TCXO/Crystal. *Maxim/Dallas.* http://pdf1.alldatasheet.com/datasheet-pdf/view/112132/DALLAS/DS3231.html (11.05.2017).

**34.** MCP3425A0T-E/CH - MICROCHIP - Analogue to Digital Converter, 16 bit, 15 SPS, Single, 2.7 V, 5.5 V, SOT-23. *Farnell element14 Eesti.* http://ee.farnell.com/microchip/mcp3425a0t-e-ch/ic-16-bit-adc-5v-sot-23-6/dp/1578433 (11.05.2017).

**35 .** MCP3425 16-Bit Analog-to-Digital Converter with I2 C Interface and On-Board Reference. *Microchip Technology Inc.* http://www.farnell.com/datasheets/525051.pdf?_ga=2.66339822.1566411666.1495452331-226024234.1479828035 (11.05.2017).

**36. Steffes, M.** (2006). Design Methodology for MFB Filters in ADC Interface Applications, Application Report SBOA114–February 2006, *Texas Instruments*, 32 lk.

**37 . Zumbahlen, H.** (2012). Mini Tutorial MT-220 Multiple Feedback Filters, *Analog Devices*, 3 lk.

**38. Farnell element14 Eesti.** http://ee.farnell.com/ (11.05.2017).

**39.** Raspberry Pi 3 mudel B moodul 1.2GHz 1GB. *Oomipood.* https://www.oomipood.ee/product/2525225_raspberry_pi_3_mudel_b_moodul_1_2ghz_1gb?q=Raspberry%20Pi%203 (11.05.2017).

**40.** 1PCS DS3231 AT24C32 IIC Precision RTC Real Time Clock Memory Module For Arduino new original-in Integrated Circuits from Electronic Components. *Alibaba Group.* https://www.aliexpress.com/item/DS3231-AT24C32-IIC-High-Precision-RTC-Module-Clock-Timer-Memory-Module/32670041263.html?spm=2114.13010308.0.0.irBfUT (11.05.2017).

**41.** 12GA DC 6V 100RPM Miniature Electric Reduction Gear Motor Metal Gearbox for RC Robot Model Toy DIY engine Camera Motor-in DC Motor from Home Improvement. *Alibaba Group.* https://www.aliexpress.com/item/12GA-DC-6V-100RPM-Miniature-Electric-Reduction-Gear-Motor-Metal-Gearbox-for-RC-robot-model-Toy/32591728713.html?spm=2114.13010308.0.0.9KbYeC (11.05.2017).

# LISAD

# LISA A

**Tabel 1.** Kaheksa olemasoleva jaoturi võrdlus [7]

| Jaoturi nimi | **Petsafe simply feed** | **Csf-3 super feeder** | **Pet feedster** | **Lusmo automatic pet feeder** | **Crown majestic (diamond series v 3)** | **Petnet smart feeder** | **Cat mate c3000** | **Wireless whiskers pet feeder** |
|---|---|---|---|---|---|---|---|---|
| Jaoturi pilt |  |  |  |  |  |  |  |  |
| Pikkus<br>Laius<br>Kõrgus, cm | 34<br>26<br>40 | 28<br>25<br>38 | 35<br>38<br>43 | 35<br>23<br>33 | -<br>-<br>- | 38<br>23<br>38 | 20<br>20<br>36 | 40<br>23<br>40 |
| Aluse pindala, $cm^2$ | 884 | 700 | 1330 | 805 | - | 874 | 400 | 920 |
| Mass, kg | 2,9 | 4,0 | 4,5 | 1,5 | 1,7 | 3,1 | 0,9 | 14 |
| Mahtuvus, L | 6 | 1,18 | 1,12 | 0,25 | 0,5 | 0,5 - 0,75 | 0,75 | 0,55 |
| Taimer | Digitaalne | Digitaalne | Digitaalne | LCD | LCD | Digitaalne | LCD | LCD |

**Tabel 1.** Kaheksa olemasoleva jaoturi võrdlus (jätk)

| Toidu tüüp ning annused | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| Toidu tüüp | Kuiv | Kuiv | Kuiv ja niiske | Kuiv | Kuiv | Kuiv | Kuiv | Kuiv |
| Toidupala suurus | Enamus suurusi | 6,35 mm ümar | Reguleeritav vastavalt toidu suurusele | Mitte suurem kui 15,24 mm | Reguleeritav vastavalt toidu suurusele | 56- 283 g Ümartoit | 283 g | 222 g |
| Toiteallikas ning varundamine | | | | | | | | |
| Toiteallikas | 4 D tüüpi patareid | *AC/DC* adapter | *AC/DC* adapter või 6 D tüüpi patareid | 4 D tüüpi patareid | *AC/DC* adapter või 4 AA tüüpi patareid | *AC/DC* adapter | 4 C tüüpi patareid | *AC* adapter |
| Tagavara toide | Jah | Ei | Jah | Ei | Jah | Jah | Ei | Ei |
| Töö voolukatkestuse korral[1] | Patareil | Ei[2] | Jah, mitu kuud | Patareil | Patareil | Jah kuni 7h | Patareil | EI kuid uksed avanevad automaatselt |
| Programm säilib peale voolu katkemist | Jah, patarei toitel | Jah | Jah, kell patarei toitel | Jah, patarei toitel | Jah, patarei toitel | Jah | Jah, patarei toitel | Jah, kell ka kuupäev patarei toitel |

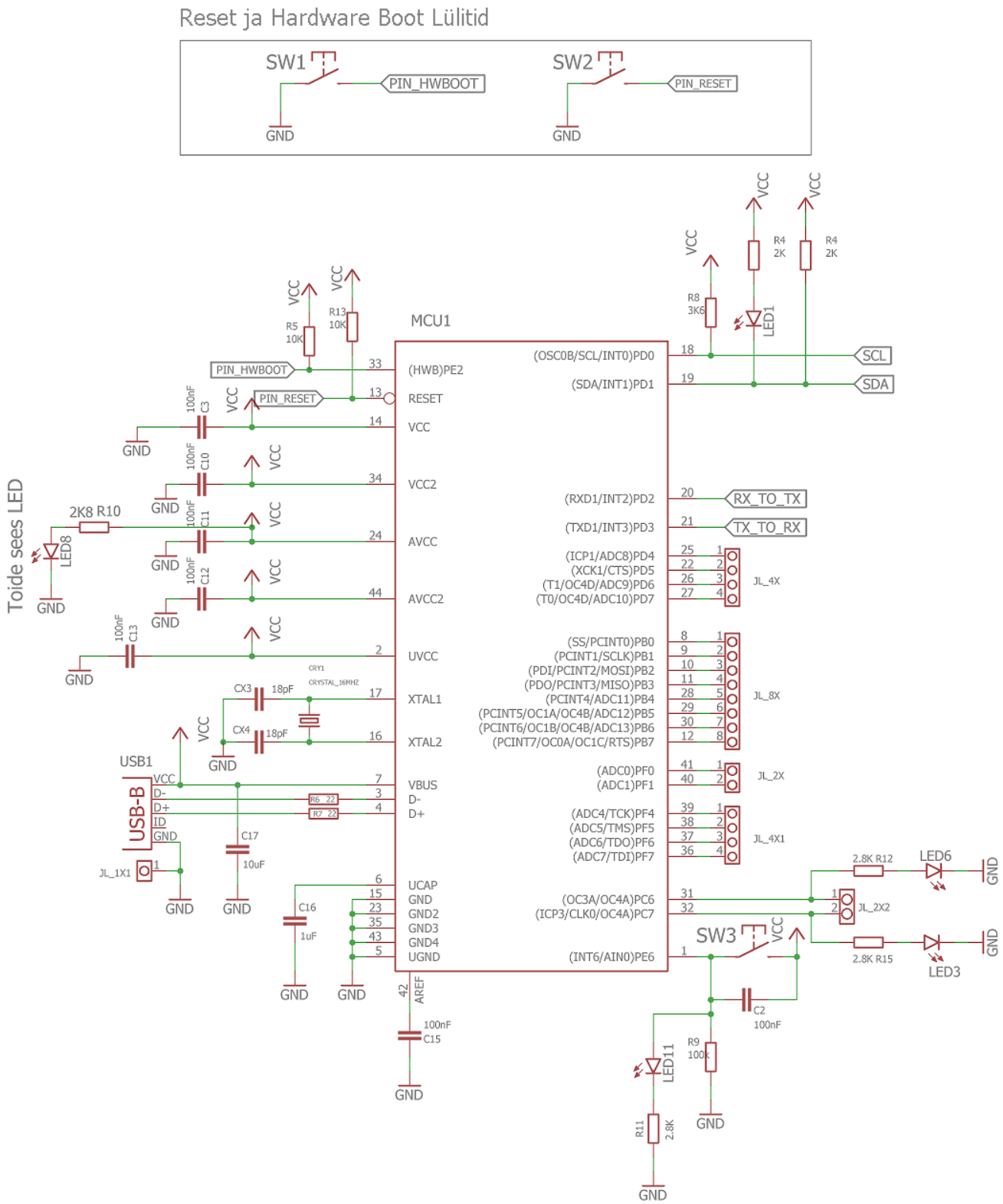
[1] Seade on võimeline töötama nii patareide kui ka adapteri pealt samaaegselt, et tagada seadme töö voolukatkestuste või tühjenenud patareide korral.

[2] Seade ei tööta voolukatkestuse korral, kuid säilitab programmi ning jätkab tööd peale voolu taastumist.

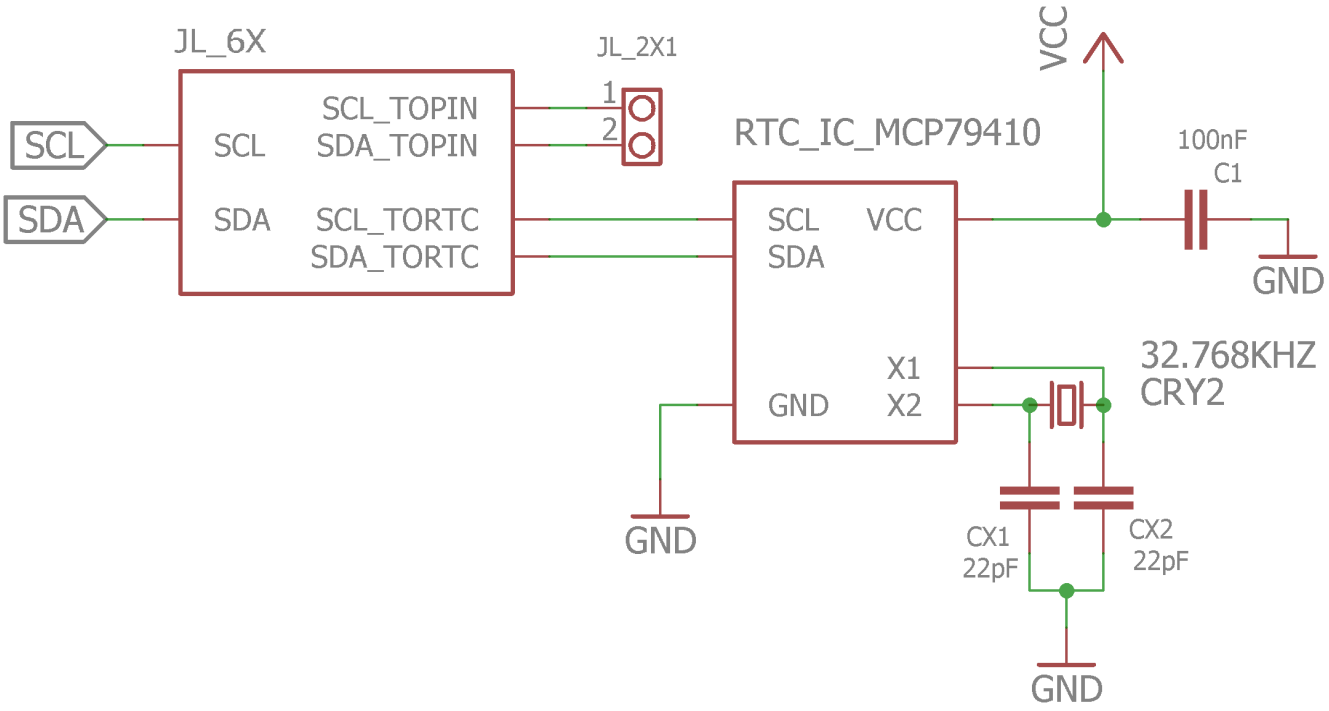
**Tabel 1.** Kaheksa olemasoleva jaoturi võrdlus (jätk)

| Toidu tüüp ning annused | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| Seadistuse salvestus patarei vahetusel | Ei | Ei | Jah, välja arvatud kell | Ei | Ei | Ei | Jah 1 minuti jooksul | Jah , välja arvatud kell |
| Patarei eluea indikaator | Jah | Ei | Jah | Jah | Jah | Jah | Jah | Jah |
| Programeeritavad tunnused | | | | | | | | |
| Söögikordi päevas | 1-12 | 1-8 | 1-5 | 1-3 | 1-4 | Sõltuvalt konteineri suurusest | 1-3 | 1-24 |
| Portsjoni suurus, L | Kuni 1 | Kuni 0,5 | Kuni 1,5 | Kuni 0,25 | Kuni 0,5 | Kuni 0,25 | - | Kuni 0,18 |
| Teised erisused | | | | | | | | |
| Hääle salvestus | Ei | Ei | Ei | Ei | Jah | Ei | Ei | Tervitab iga looma nime pidi |
| Automati-seerimine | Ei | Jah | Ei | Ei | Ei | Kontrollitav telefoni teel | Ei | Ei |
| Mitme looma programmi tugi | Ei | Ei | Ei | Ei | Ei | Ei | Ei | Jah |
| Ligikaudne ind | 123 € | 142 € | 152 € | 93 € | 66 € | 142 € | 47 € | 161 € |

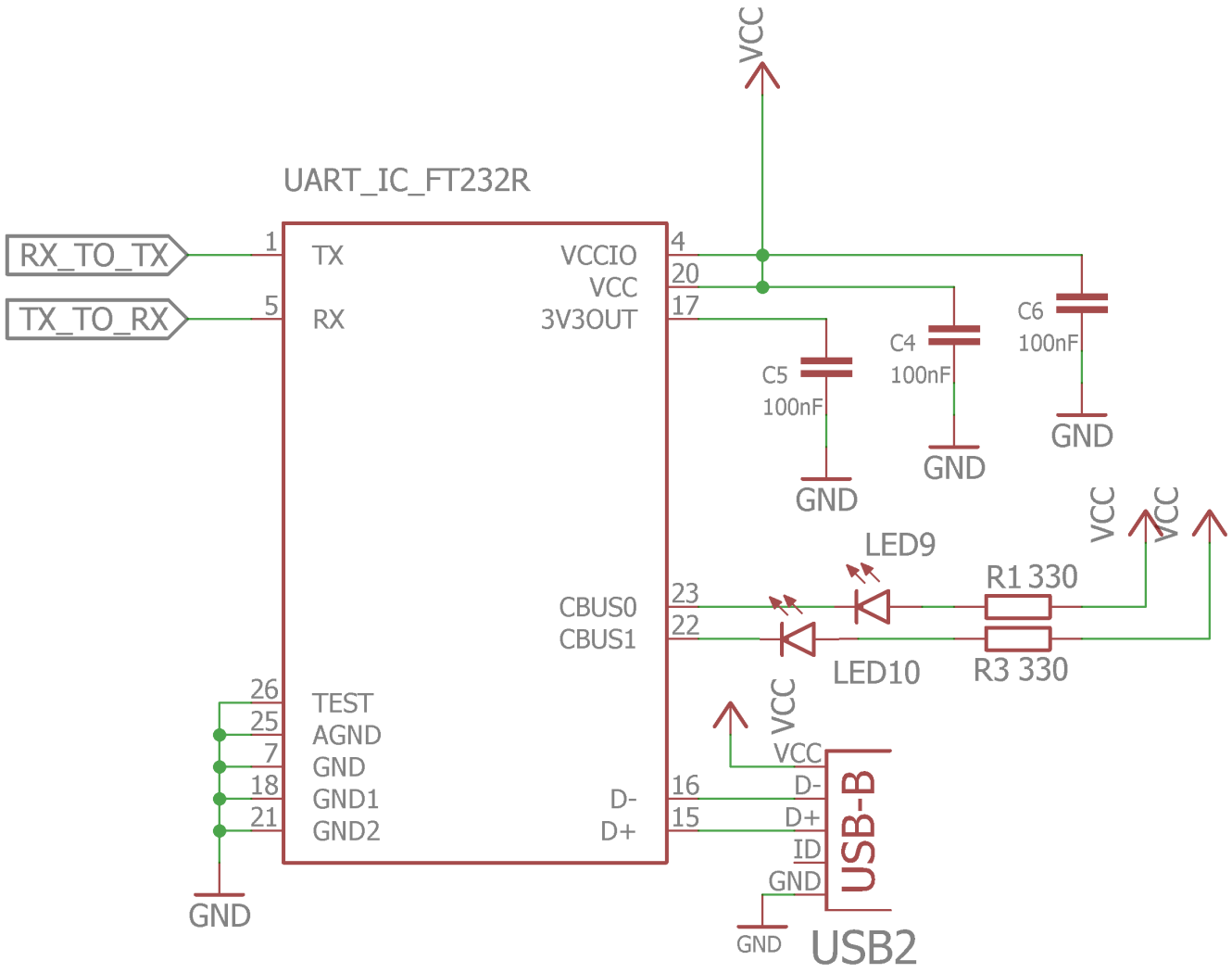
# LISA B



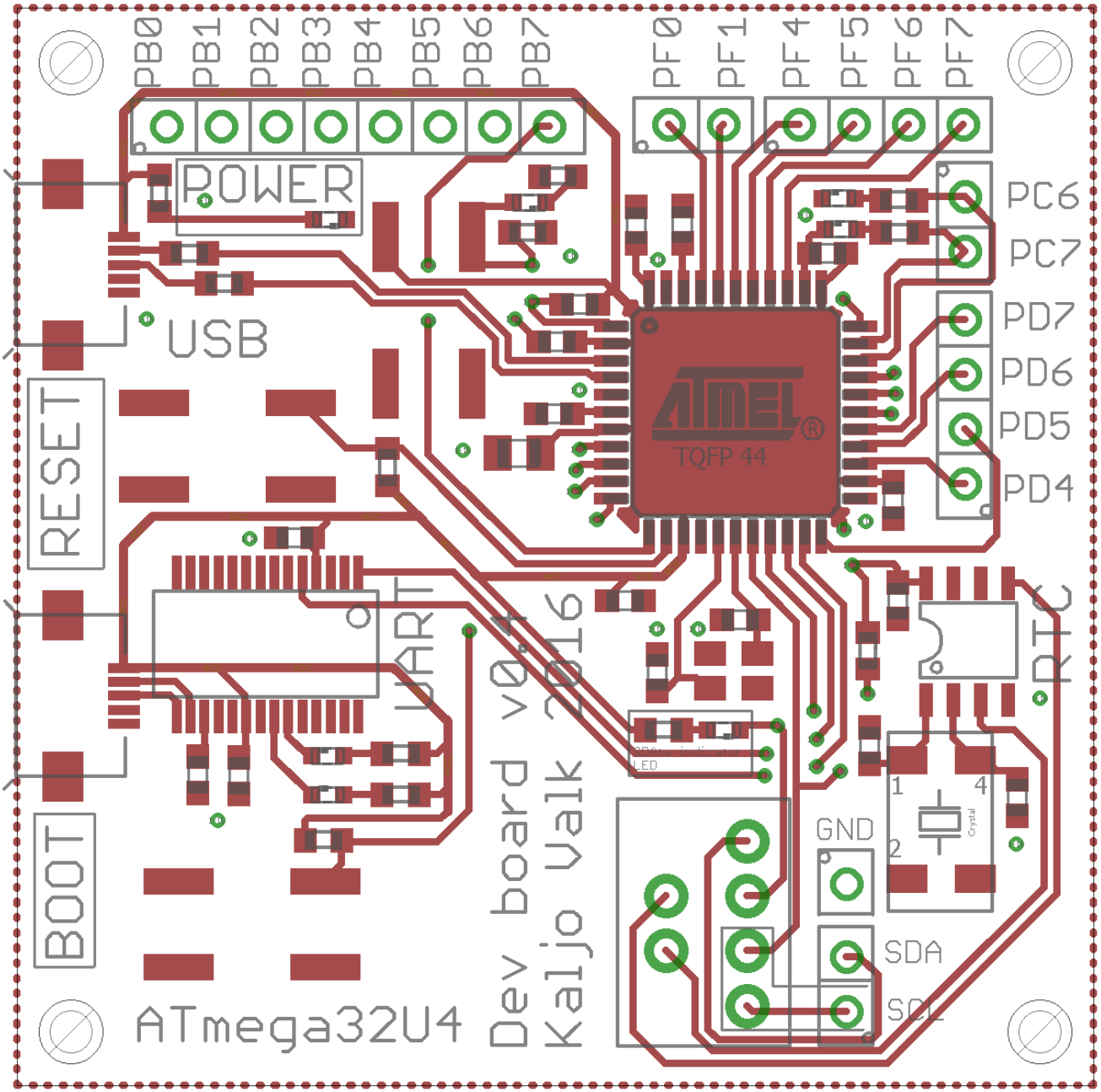

| | | | |
|---|---|---|---|
| *Teostas* | *Kaljo Valk* | *Nimetus:* | |
| *Kontrollis* | *Erkki Jõgi* | *Mikrokontrolleri elektroonika põhimõtte skeem* | |
| *Kinnitas* | *Erkki Jõgi* | | |
| *EMÜ TS-TN* | | *Leht:* *1/7* | *Tähis:* *TN 17/130261 B 01 00 S* |




| | | | |
|---|---|---|---|
| *Teostas* | *Kaljo Valk* | *Nimetus:* | |
| *Kontrollis* | *Erkki Jõgi* | *Reaalajakella elektrooniline põhimõtte skeem* | |
| *Kinnitas* | *Erkki Jõgi* | | |
| *EMÜ TS-TN* | | *Leht:* *2/7* | *Tähis:* *TN 17/130261 B 01 01 S* |

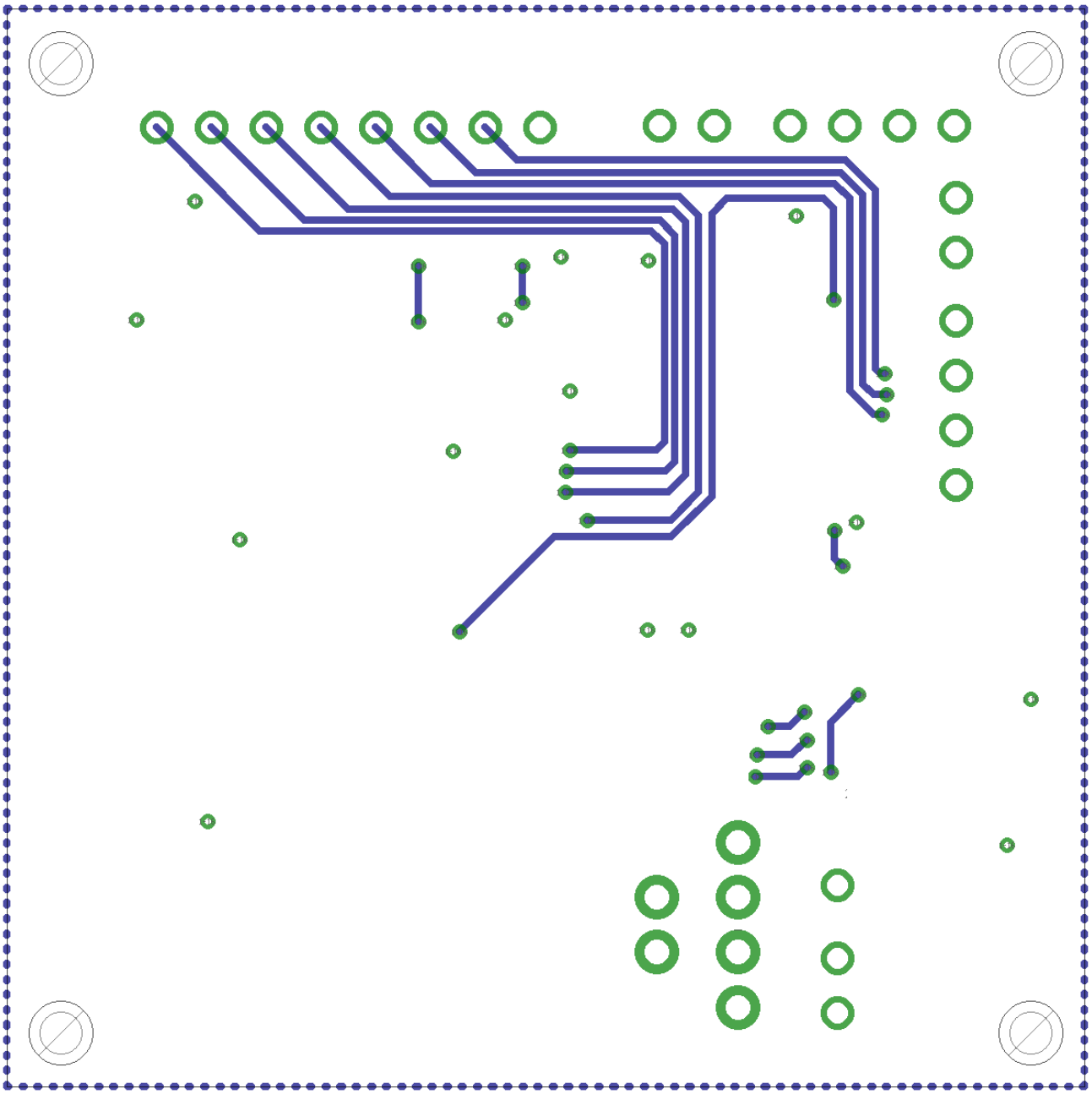



| | | | | |
|---|---|---|---|---|
| *Teostas* | *Kaljo Valk* | *Nimetus:* | | |
| *Kontrollis* | *Erkki Jõgi* | *UART mooduli elektrooniline põhimõtte skeem* | | |
| *Kinnitas* | *Erkki Jõgi* | | | |
| *EMÜ TS-TN* | | *Leht:* *3/7* | *Tähis:* *TN 17/130261 B 01 02 S* | |




*Trükkplaadi füüsilised mõõtmed: 50 x 50 mm*

| | | | | |
|---|---|---|---|---|
| *Materjal:* | | *Näitamata piirhälbed:* | *Mass* | *Mõõt 4:1* |
| *Teostas* | *Kaljo Valk* | *Nimetus:* *Arendusplaadi ühenduste skeem Ülemine kiht* | | |
| *Kontrollis* | *Erkki Jõgi* | | | |
| *Kinnitas* | *Erkki Jõgi* | | | |
| *EMÜ TS-TN* | | *Leht:* *4/7* | *Tähis:* *TN 17/130261 B 01 03 S* | |



*Trükkplaadi füüsilised mõõtmed: 50 x 50 mm*

| | | | | |
|---|---|---|---|---|
| *Materjal:* | | *Näitamata piirhälbed:* | *Mass* | *Mõõt 4:1* |
| *Teostas* | *Kaljo Valk* | *Nimetus:* *Arendusplaadi ühenduste skeem Alumine kiht* | | |
| *Kontrollis* | *Erkki Jõgi* | | | |
| *Kinnitas* | *Erkki Jõgi* | | | |
| *EMÜ TS-TN* | | *Leht:* *5/7* | *Tähis:* *TN 17/130261 B 01 04 S* | |

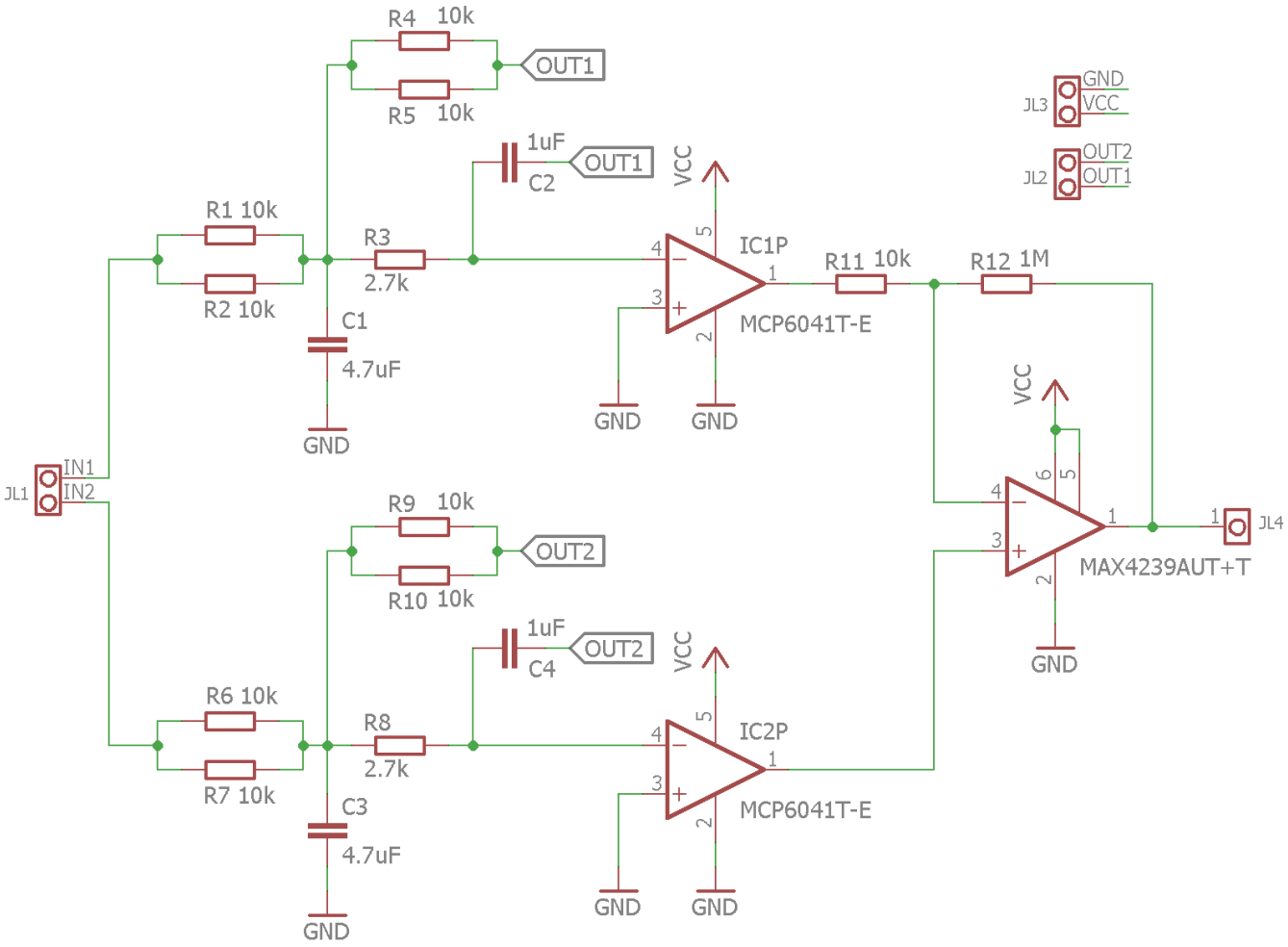
| | | | |
|---|---|---|---|
| MCU1 | ATMEGA32U4 | 1 | |
| R10-15 | 2,8 k takisti | 3 | |
| R9 | 100 k takisti | 1 | |
| R8 | 3,6 k takisti | 1 | |
| R6,R7 | 22 k takisti | 1 | |
| R4 | 2 k takisti | 2 | |
| R1,R3 | 330 takisti | 2 | |
| CX3,CX4 | 18 pF kondensaator | 2 | |
| CX1,CX2 | 22 pF kondensaator | 2 | |
| C17 | 10 uF kondensaator | 1 | |
| C16 | 1 uF kondensaator | 1 | |
| R5,13 | 10 k takisti | 2 | |
| C10-15 | 100 nF kondensaator | 5 | |
| C1-6 | 100 nF kondensaator | 6 | |
| USB1,2 | Micro USB pistikupesa | 2 | |
| LED 11 | Valgusdiood | 1 | |
| LED 10 | Valgusdiood | 1 | |
| LED 9 | Valgusdiood | 1 | |
| LED 8 | Valgusdiood | 1 | |
| LED 6 | Valgusdiood | 1 | |
| LED 3 | Valgusdiood | 1 | |
| LED 1 | Valgusdiood | 1 | |
| Tähis | Nimetus | Hulk | Märkus |

| | | | |
|---|---|---|---|
| Tellija | | Objekt, seade<br>Arendusplaat | |
| Teostas | Kaljo Valk | Nimetus:<br>Arendusplaadi elektroonikakomponentide nimistu | |
| Kontrollis | Erkki Jõgi | | |
| Kinnitas | Erkki Jõgi | | |
| EMÜ TS-TN | | Leht: 6/7 | Tähis: TN 17/130261 B 01 05 S |

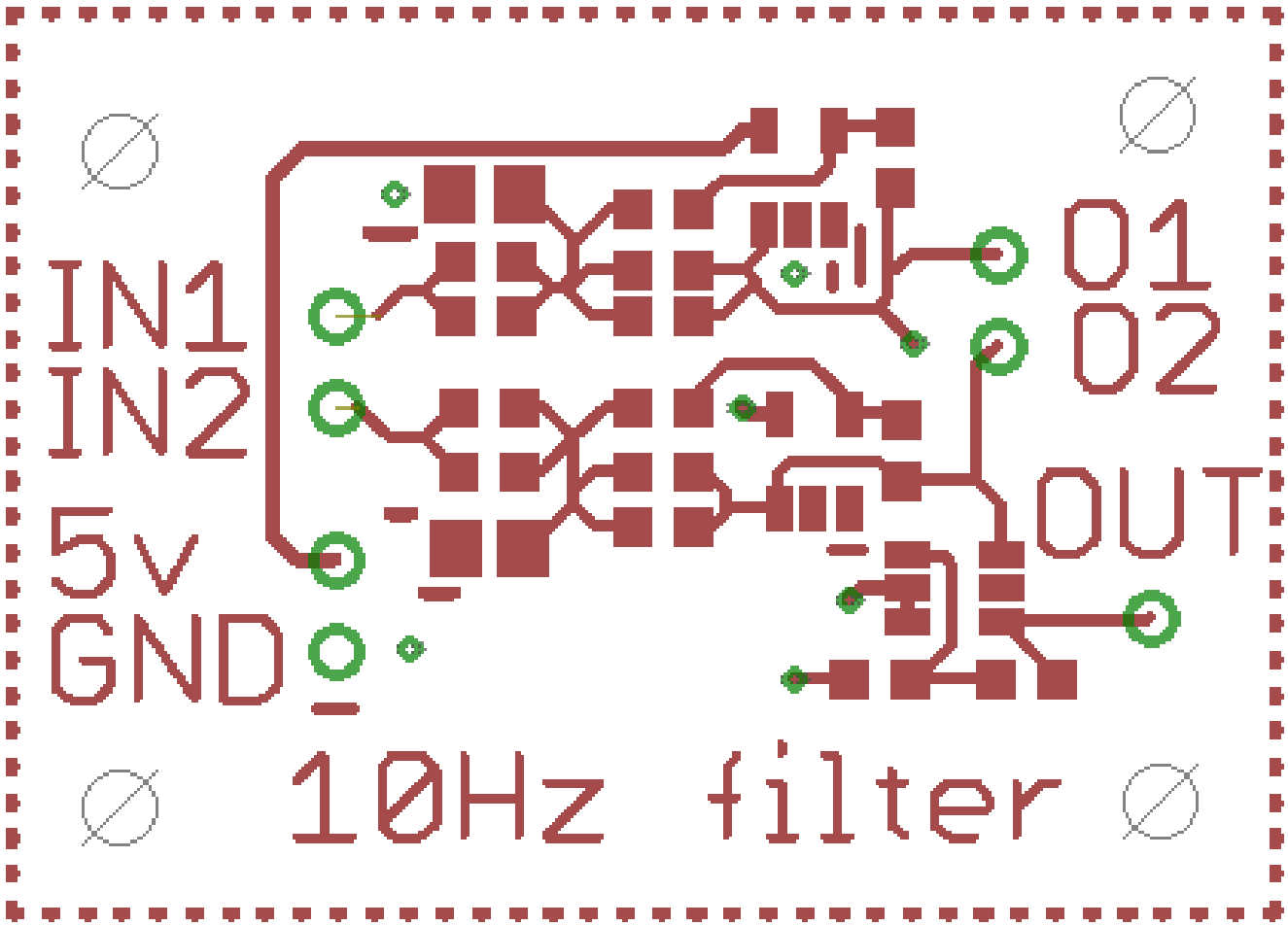
| | | | |
|---|---|---|---|
| JL_8x | Pistik ühendused | 1 | |
| JL_6x | Pistik ühendused | 1 | |
| JL_4x | Pistik ühendused | 2 | |
| JL_2x | Pistik ühendused | 3 | |
| JL_1x | Pistik ühendused | 1 | |
| RTC | RTC_IC_MCP79410 reaalajakell | 1 | |
| SW1-3 | Lülitid | 3 | |
| UART | UART_IC_FT232R | 1 | |
| CRY2 | 32.768 KHz kristall | 1 | |
| CRY1 | 16 MHz kristall | 1 | |
| Tähis | Nimetus | Hulk | Märkus |

| | | | |
|---|---|---|---|
| Tellija | | Objekt, seade<br>Arendusplaat | |
| Teostas | Kaljo Valk | Nimetus:<br>Arendusplaadi elektroonilinkakomponentide nimistu | |
| Kontrollis | Erkki Jõgi | | |
| Kinnitas | Erkki Jõgi | | |
| EMÜ TS-TN | | Leht:<br>7/7 | Tähis:<br>TN 17/130261 B 01 06 S |

# LISA C

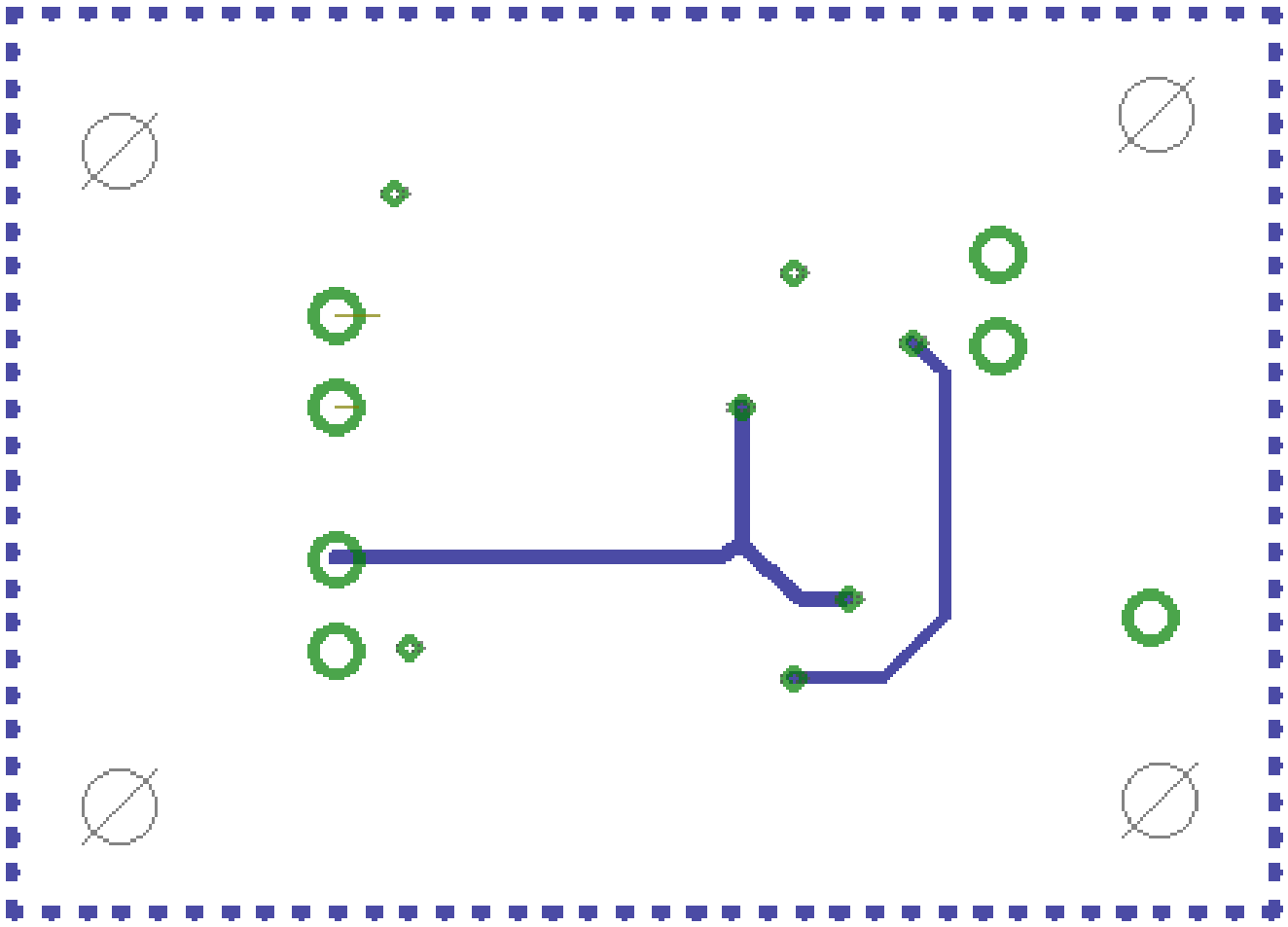



| | | | |
|---|---|---|---|
| *Teostas* | *Kaljo Valk* | *Nimetus:* | |
| *Kontrollis* | *Erkki Jõgi* | *Filtrite elektroonika põhimõtte skeem* | |
| *Kinnitas* | *Erkki Jõgi* | | |
| *EMÜ TS-TN* | | *Leht:* *1/4* | *Tähis:* *TN 17/130261 C 01 00 S* |




*Trükkplaadi füüsilised mõõtmed: 35 x 25 mm*

| | | | | |
|---|---|---|---|---|
| *Materjal:* | | *Näitamata piirhälbed:* | *Mass* | *Mõõt 5:1* |
| *Teostas* | *Kaljo Valk* | *Nimetus:* *Filtrite ühenduste skeem Ülemine kiht* | | |
| *Kontrollis* | *Erkki Jõgi* | | | |
| *Kinnitas* | *Erkki Jõgi* | | | |
| *EMÜ TS-TN* | | *Leht:* *2/4* | *Tähis:* *TN 17/130261 C 01 01 S* | |



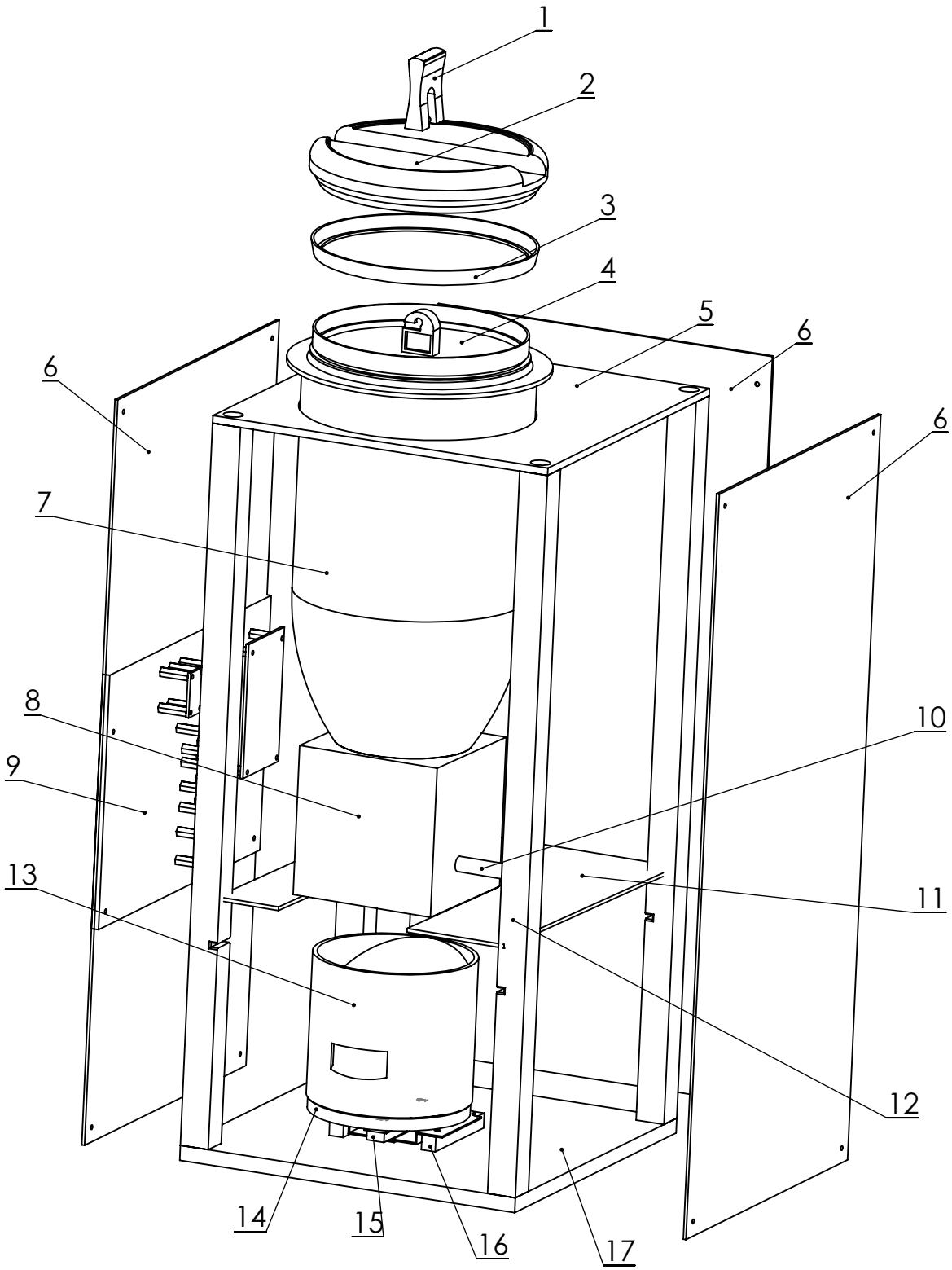
*Trükkplaadi füüsilised mõõtmed: 35 x 25 mm*

| | | | | |
|---|---|---|---|---|
| *Materjal:* | | *Näitamata piirhälbed:* | *Mass* | *Mõõt 5:1* |
| *Teostas* | *Kaljo Valk* | *Nimetus:* *Filtrite ühenduste skeem Alumine kiht* | | |
| *Kontrollis* | *Erkki Jõgi* | | | |
| *Kinnitas* | *Erkki Jõgi* | | | |
| *EMÜ TS-TN* | | *Leht:* *3/4* | *Tähis:* *TN 17/130261 C 01 02 S* | |

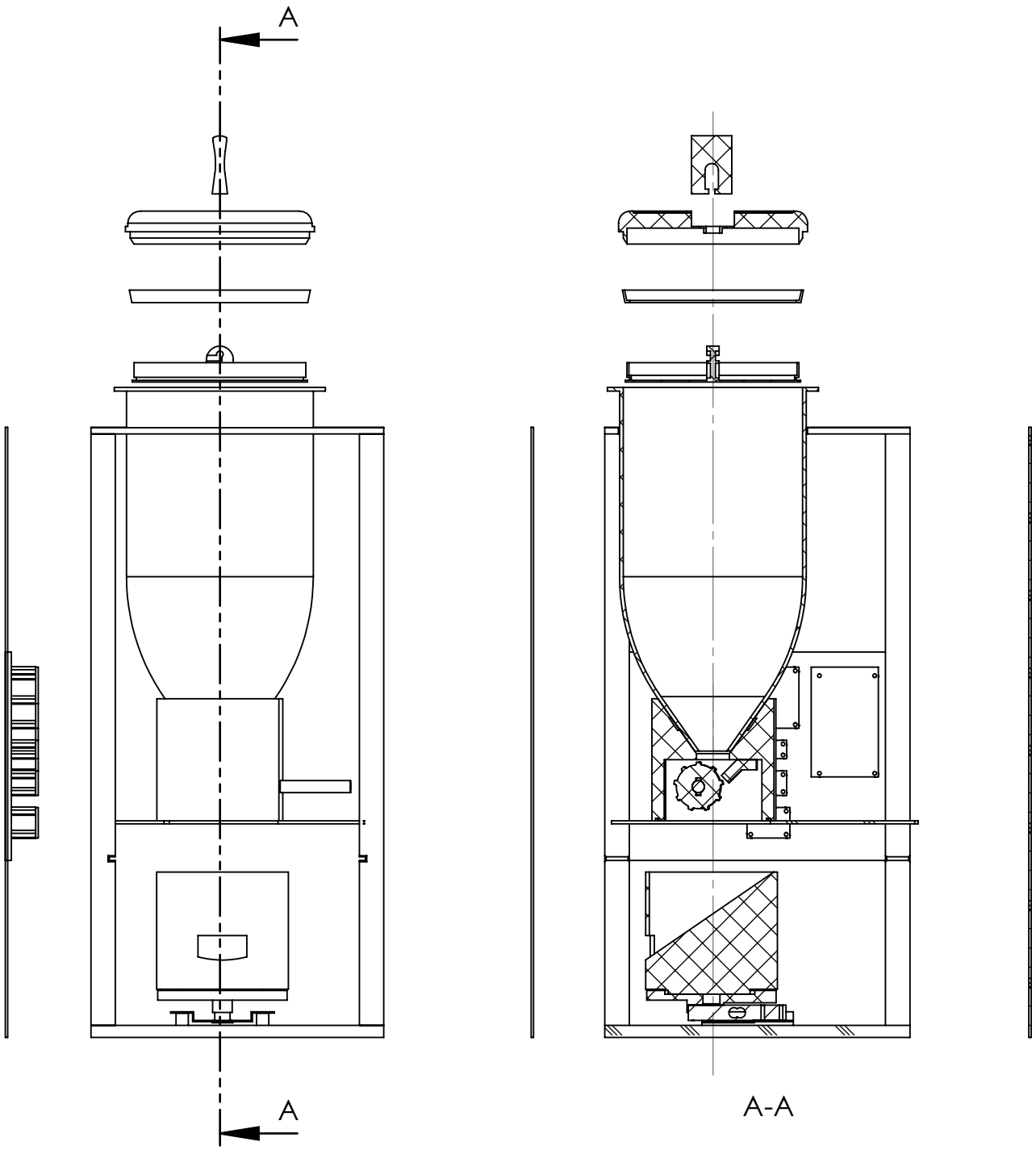
| | | | |
|---|---|---|---|
| R12 | 1 M takisti | 1 | |
| JL1-4 | Jumper ühendused | 4 | |
| IC3 | MAX4239AUT+T operatsioonivõimendi | 1 | |
| IC1,IC2 | MCP6041T-E operatsioonivõimendi | 2 | |
| C2,C4 | 1 uF kondensaator | 1 | |
| C1,C3 | 4,7 uF kondensaator | 1 | |
| R9-11 | 10 k takisti | 2 | |
| R8 | 2,7 k takisti | 1 | |
| R4-7 | 10 k takisti | 4 | |
| R3 | 2,7 k takisti | 1 | |
| R1,R2 | 10 k takisti | 2 | |
| Tähis | Nimetus | Hulk | Märkus |

| | | | |
|---|---|---|---|
| Tellija | | Objekt, seade: Filtrite plaat | |
| Teostas | Kaljo Valk | Nimetus: Elektroonikakomponentide nimistu | |
| Kontrollis | Erkki Jõgi | | |
| Kinnitas | Erkki Jõgi | | |
| EMÜ TS-TN | | Leht: 4/4 | Tähis: TN 17/130261 C 01 03 S |

# LISA D




| | | | | |
|---|---|---|---|---|
| *Materjal:* | | *Näitamata piirhälbed:* | *Mass 4,1 kg* | *Mõõt 1:4* |
| *Teostas* | *Kaljo Valk* | *Nimetus:* *Seadme koost* | | |
| *Kontrollis* | *Erkki Jõgi* | | | |
| *Kinnitas* | *Erkki Jõgi* | | | |
| *EMÜ TS-TN* | | *Leht:* *1/4* | *Tähis:* *TN 17/130261 D1 01 00 K* | |



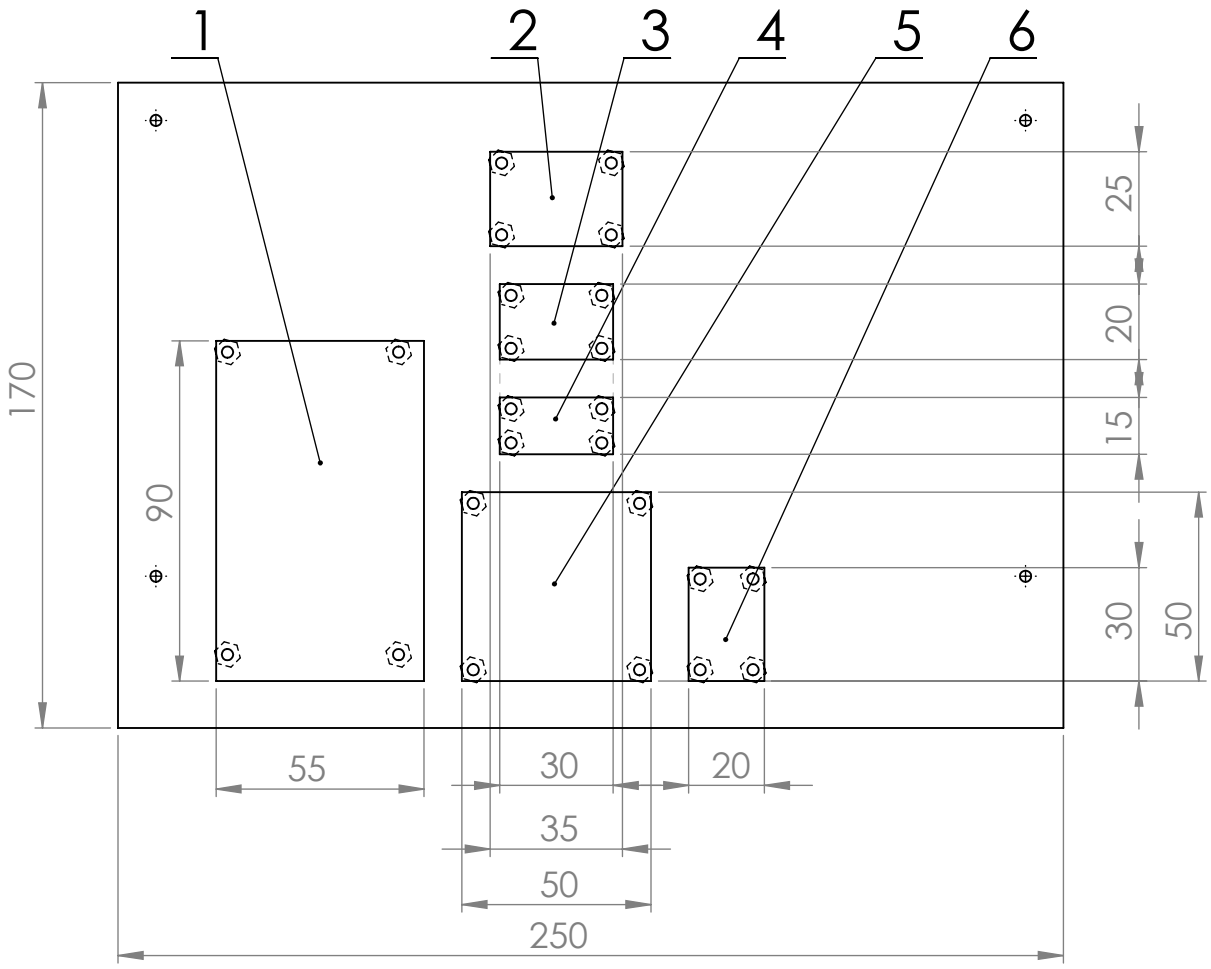

| | | | | |
|---|---|---|---|---|
| Materjal: | | Näitamata piirhälbed: | Mass 4,1 kg | Mõõt 1:5 |
| Teostas | Kaljo Valk | Nimetus: Seadme koost Läbilõige | | |
| Kontrollis | Erkki Jõgi | | | |
| Kinnitas | Erkki Jõgi | | | |
| EMÜ TS-TN | | Leht: 2/4 | Tähis: TN 17/130261 D 01 01 K | |

| Nr. | Tähis | Kirjeldus | Kogus |
|---|---|---|---|
| 1 | TN 17/130261 I 01 00 D | Kaane ülemine osa, käepide | 1 |
| 2 | TN 17/130261 I 01 02 D | Kaas, ülemine osa | 1 |
| 3 | - | Tihend | 1 |
| 4 | TN 17/130261 I 01 03 D | Kaas, alumine osa | 1 |
| 5 | TN 17/130261 J 01 00 D | Korpuse ülemine osa | 1 |
| 6 | TN 17/130261 J 01 02 D | Korpuse küljepaneelid | 3 |
| 7 | TN 17/130261 F 01 00 D | Lehter | 1 |
| 8 | TN 17/130261 G 01 00 K | Kuup | 1 |
| 9 | TN 17/130261 D 01 03 K | Paneel elektroonikakomponentidega | 1 |
| 10 | - | Võll | 1 |
| 11 | TN 17/130261 J 01 04 D | Vaheplants | 1 |
| 12 | TN 17/130261 J 01 05 D | Küljetoed | 1 |
| 13 | TN 17/130261 H 01 00 D | Kaalumiskauss | 1 |
| 14 | TN 17/130261 E 01 01 D | Tensoanduri ülemine kinnitus | 1 |
| 15 | - | Tensoandur | 1 |
| 16 | TN 17/130261 E 01 00 D | Tensoanduri alumine kinnitus | 1 |
| 17 | TN 17/130261 J 01 01 D | Korpuse alus | 4 |

| Materjal: | Näitamata piirhälbed: | Mass | Mõõt |
|---|---|---|---|
| | | | |

| | | |
|---|---|---|
| Teostas | Kaljo Valk | Nimetus: Seadme koost<br>Komponentide loetelu |
| Kontrollis | Erkki Jõgi | |
| Kinnitas | Erkki Jõgi | |

| | | |
|---|---|---|
| EMÜ TS-TN | Leht: 3/4 | Tähis: TN 17/130261 D 01 02 K |



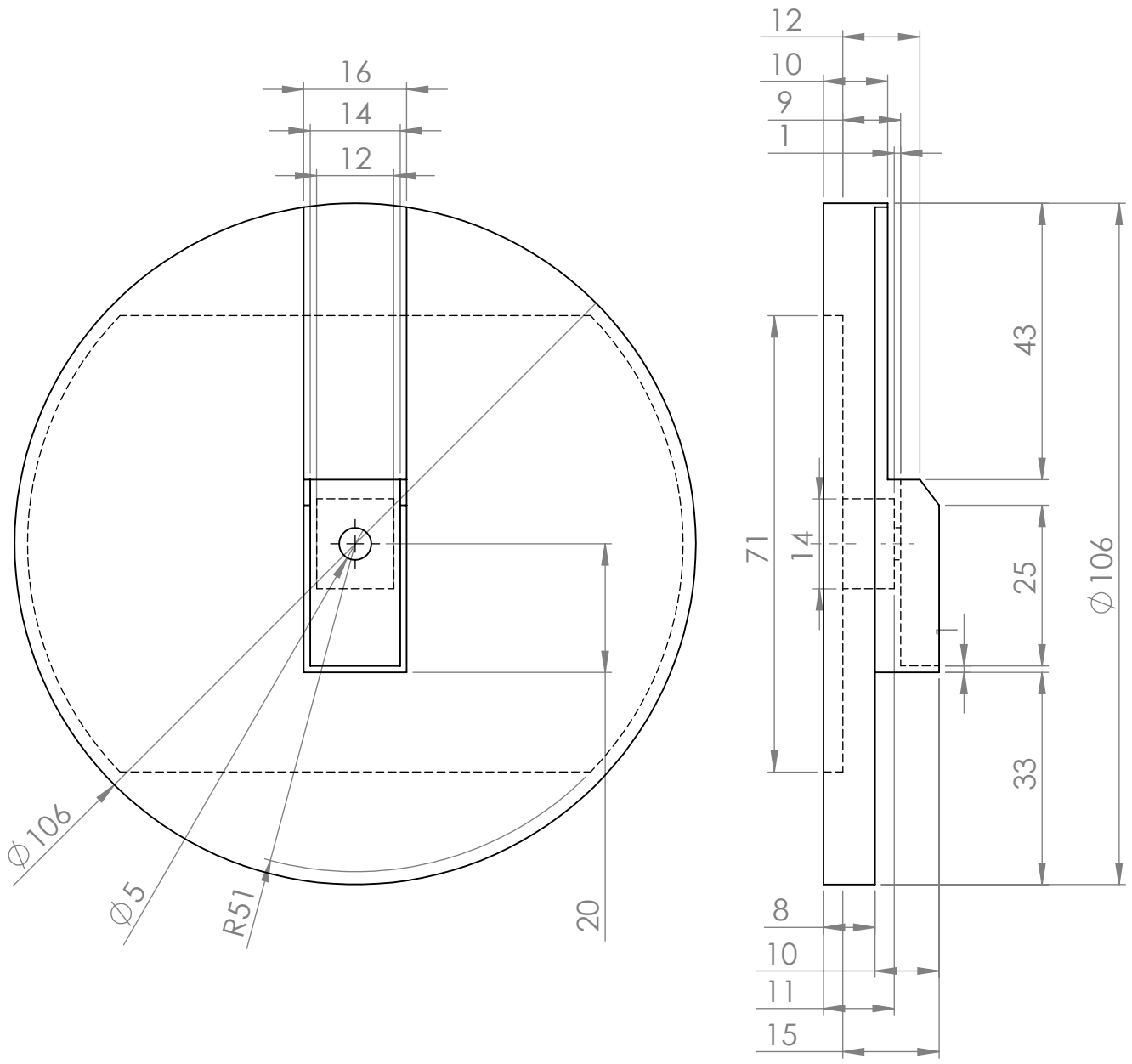

| Nr. | Märkus | Kirjeldus | Kogus |
|---|---|---|---|
| 1 | | Raspberry Pi 3 Model B | 1 |
| 2 | | Filtrid ja võimendi | 1 |
| 3 | | Analoog-digitaalmuundi | 1 |
| 4 | | Komponentidevahelised lülitiühendused | 1 |
| 5 | | Atmega32U4 Arendusplaat | 1 |
| 6 | | Reaalajakell | 1 |

| Materjal: | Näitamata piirhälbed: | Mass | Mõõt 1:2 |
|---|---|---|---|
| | | | |

| | | |
|---|---|---|
| Teostas | Kaljo Valk | Nimetus: Elektroonikakomponentide paiknemine korpuses |
| Kontrollis | Erkki Jõgi | |
| Kinnitas | Erkki Jõgi | |

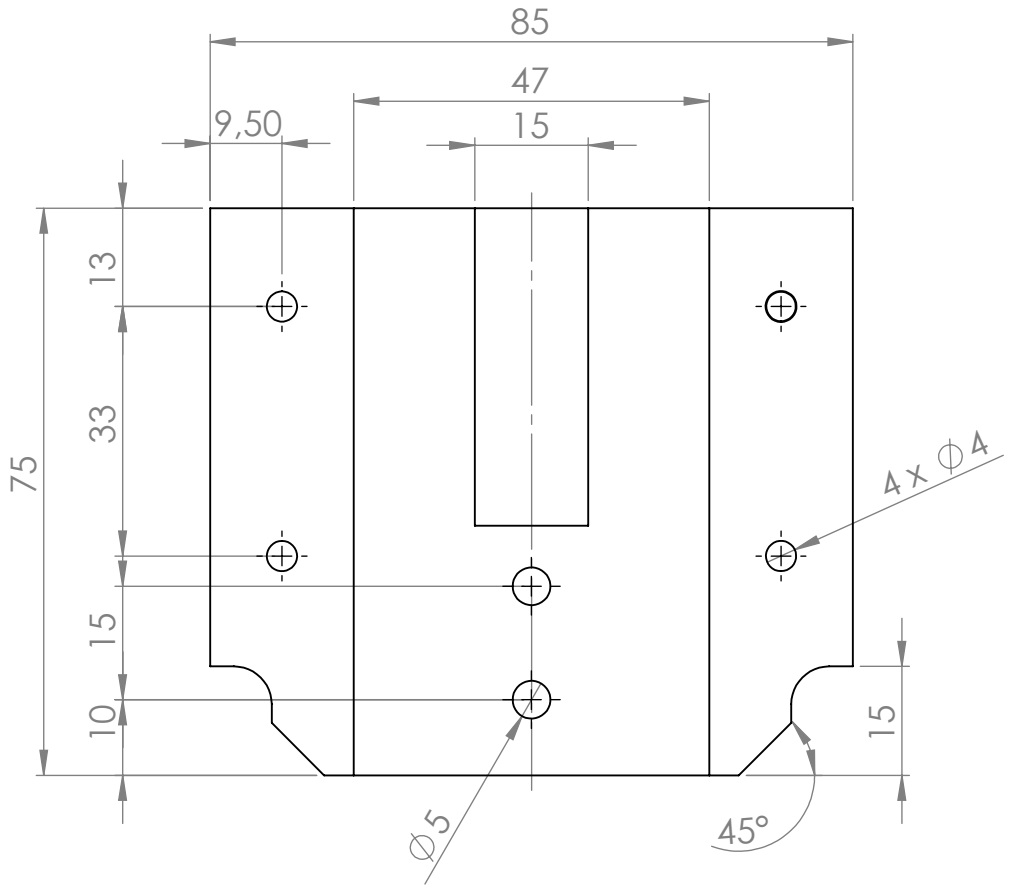
| EMÜ TS-TN | Leht: 4/4 | Tähis: TN 17/130261 D 01 03 K |
|---|---|---|

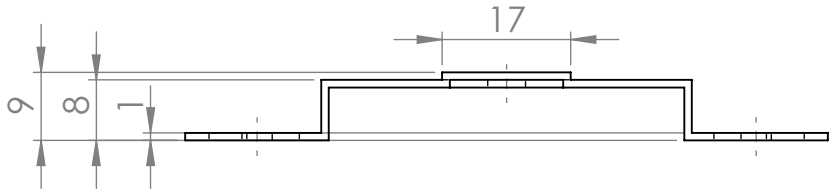
# LISA E




| | | | | |
|---|---|---|---|---|
| | *Materjal:* POM Acetal Copolymer | *Näitamata piirhälbed:* ISO 2768-m | *Mass* 64,61g | *Mõõt* 1:1 |

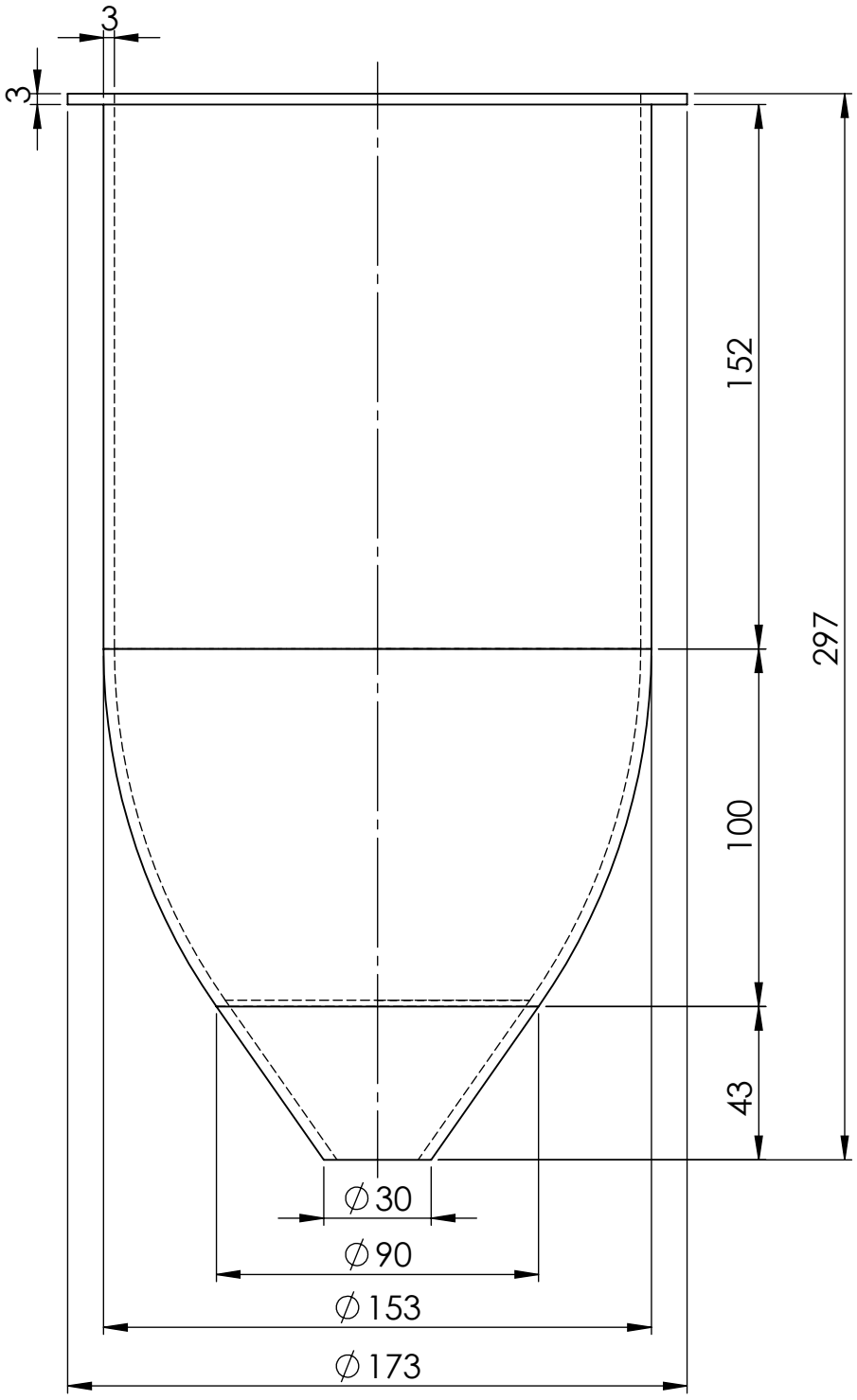
| | | | |
|---|---|---|---|
| *Teostas* | *Kaljo Valk* | *Nimetus:* Tensoanduri ülemine kinnitus | |
| *Kontrollis* | *Erkki Jõgi* | | |
| *Kinnitas* | *Erkki Jõgi* | | |
| *EMÜ TS-TN* | | *Leht:* 2/2 | *Tähis:* TN 17/130261 E 01 01 D |




| | | | | |
|---|---|---|---|---|
| | Materjal: 5052-H32 | Näitamata piirhälbed: ISO 2768-m | Mass 19,20g | Mõõt 1:1 |
| Teostas | Kaljo Valk | Nimetus: Tensoanduri alumine kinnitus | | |
| Kontrollis | Erkki Jõgi | | | |
| Kinnitas | Erkki Jõgi | | | |
| EMÜ TS-TN | | Leht: 1/2 | Tähis: TN 17/130261 E 01 00 D | |

# LISA F

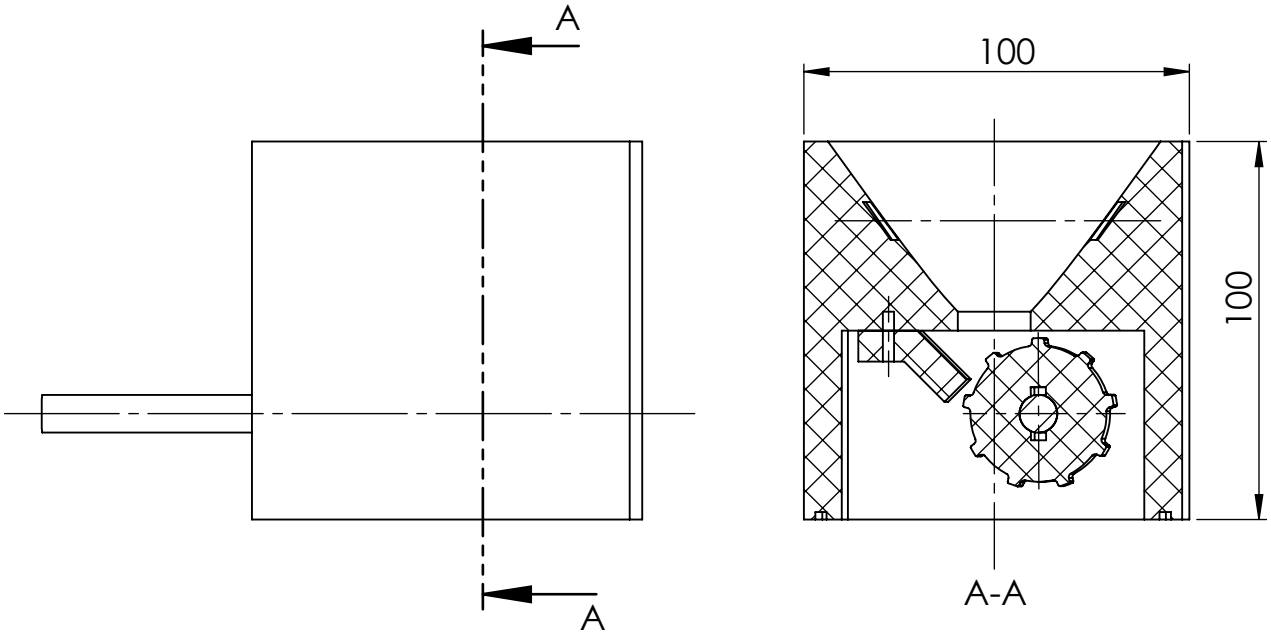



| | | | | |
|---|---|---|---|---|
| | Materjal: POM Acetal Copolymer | Näitamata piirhälbed: ISO 2768-m | Mass 544 g | Mõõt 1:2 |

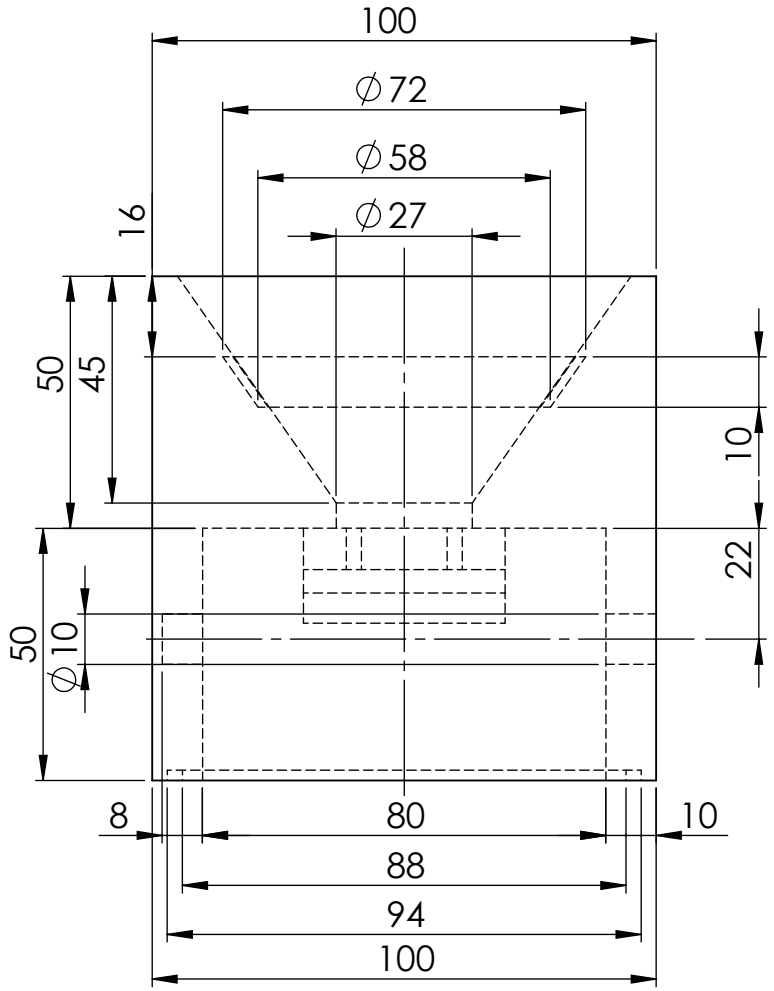
| | | |
|---|---|---|
| Teostas | Kaljo Valk | Nimetus: Lehter |
| Kontrollis | Erkki Jõgi | |
| Kinnitas | Erkki Jõgi | |

| | | |
|---|---|---|
| EMÜ TS-TN | Leht: 1/1 | Tähis: TN 17/130261 F 01 00 D |

# LISA G



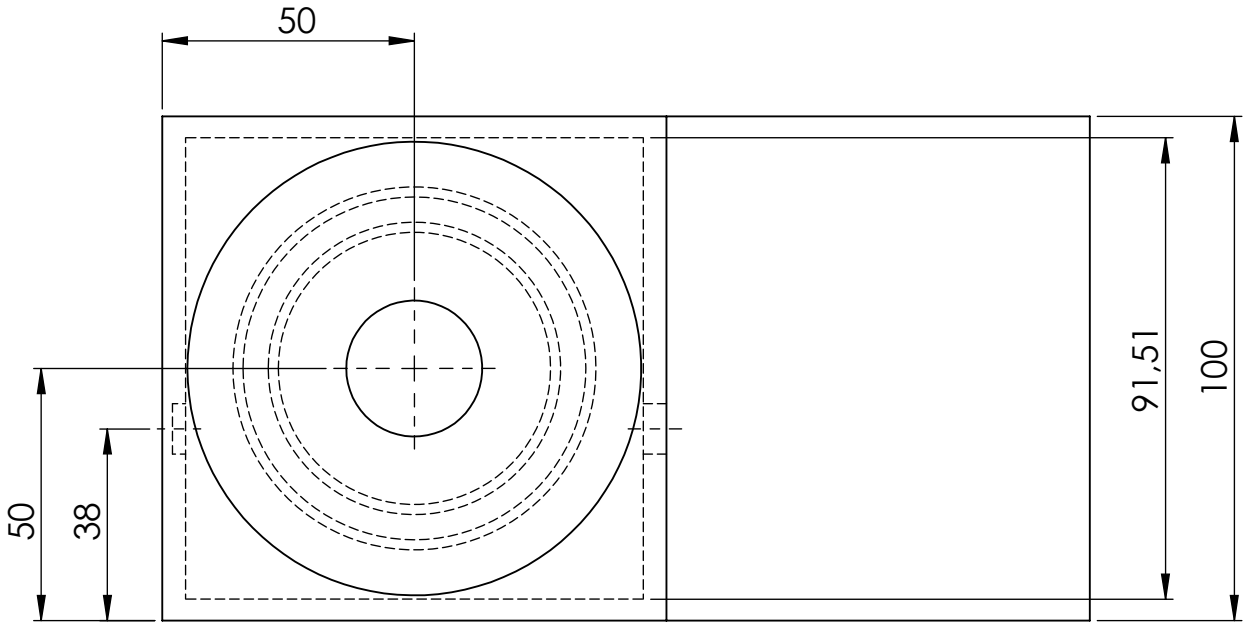

| | | | | |
|---|---|---|---|---|
| *Materjal:* | | *Näitamata piirhälbed:* | *Mass* | *Mõõt 1:2* |
| *Teostas* | *Kaljo Valk* | *Nimetus:* *Kuup* *Koostu joonis* | | |
| *Kontrollis* | *Erkki Jõgi* | | | |
| *Kinnitas* | *Erkki Jõgi* | | | |
| *EMÜ TS-TN* | | *Leht:* *1/5* | *Tähis:* *TN 17/130261 G 01 00 K* | |



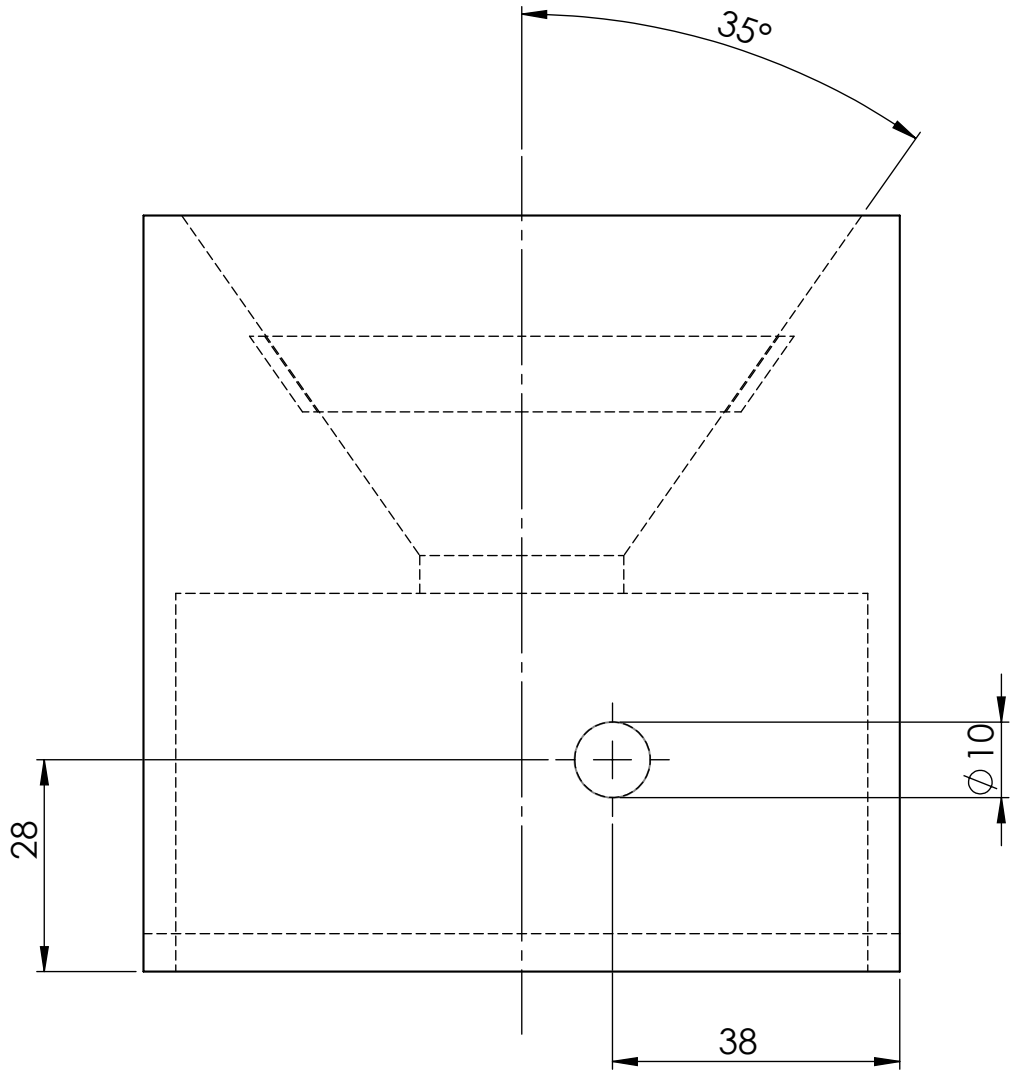

| | | | | |
|---|---|---|---|---|
| | Materjal: POM Acetal Copolymer | Näitamata piirhälbed: ISO 2768-m | Mass 676 g | Mõõt 1:1,5 |

| | | |
|---|---|---|
| Teostas | Kaljo Valk | Nimetus: Kuup Eestvaade |
| Kontrollis | Erkki Jõgi | |
| Kinnitas | Erkki Jõgi | |

| | | |
|---|---|---|
| EMÜ TS-TN | Leht: 2/5 | Tähis: TN 17/130261 G 01 01 D |

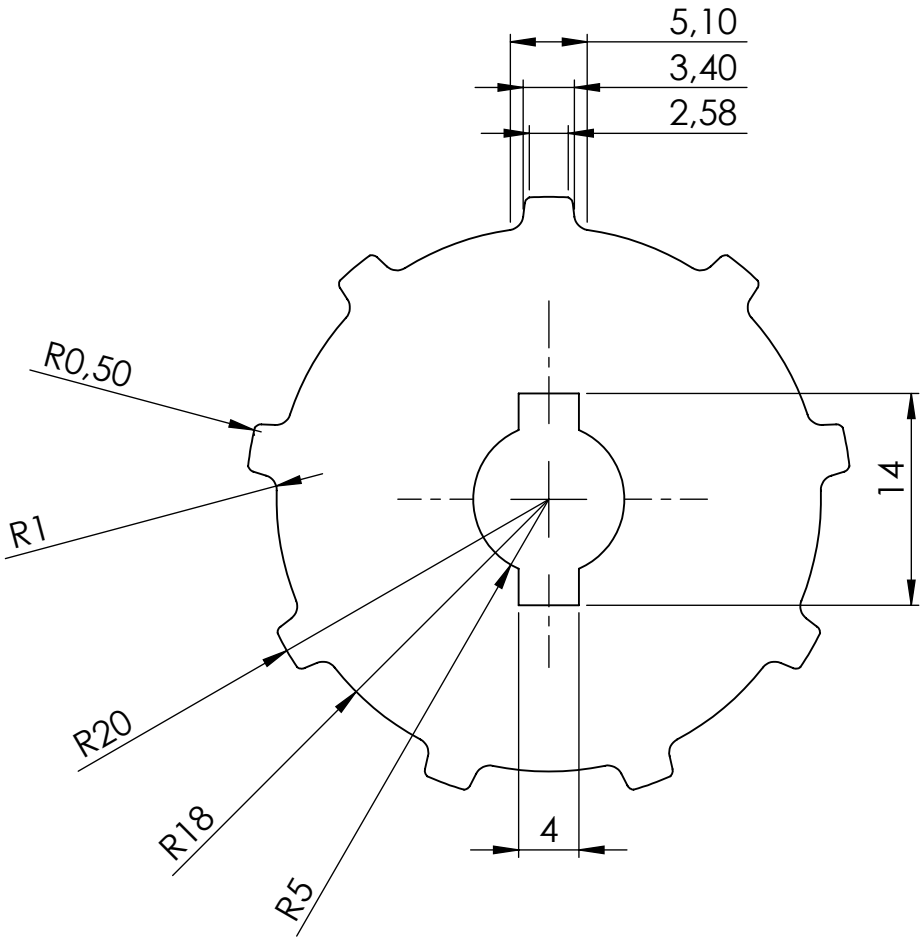



| | | | | |
|---|---|---|---|---|
| | Materjal: POM Acetal Copolymer | Näitamata piirhälbed: ISO 2768-m | Mass 676 g | Mõõt 1:1,5 |
| Teostas | Kaljo Valk | Nimetus: Kuup Pealtvaade | | |
| Kontrollis | Erkki Jõgi | | | |
| Kinnitas | Erkki Jõgi | | | |
| EMÜ TS-TN | | Leht 3/5 | Tähis: TN 17/130261 G 01 02 D | |




| | | | | |
|---|---|---|---|---|
| | Materjal: POM Acetal Copolymer | Näitamata piirhälbed: ISO 2768-m | Mass 676 g | Mõõt 1:1,5 |

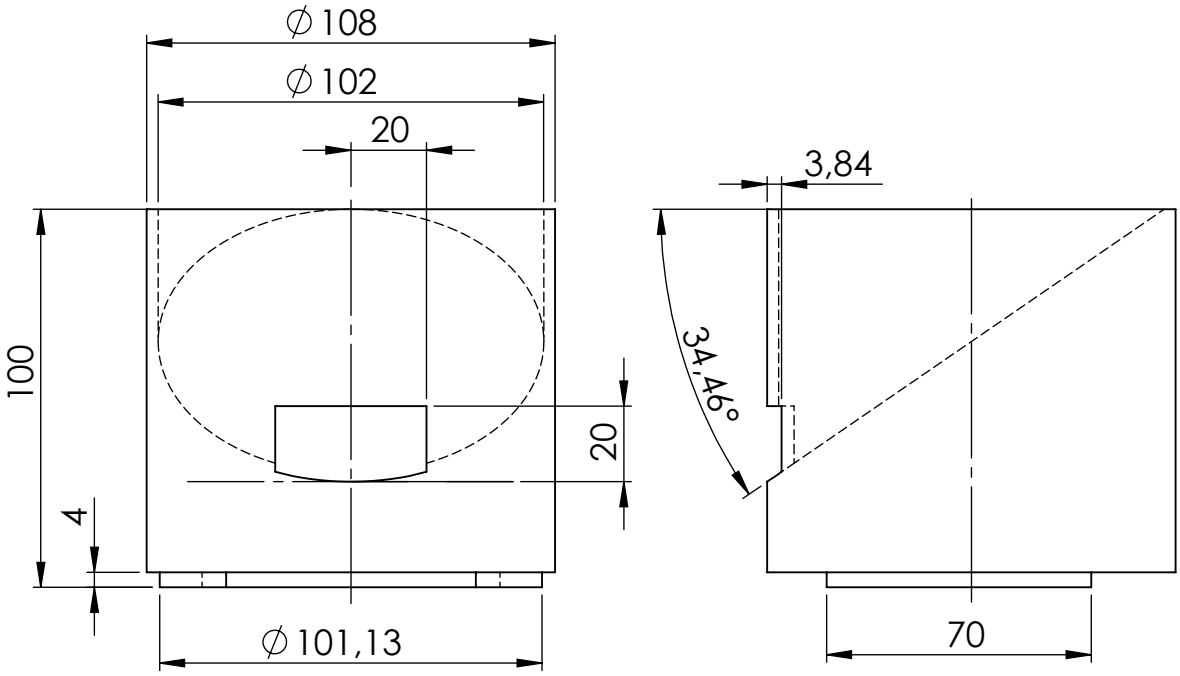
| | | | |
|---|---|---|---|
| Teostas | Kaljo Valk | Nimetus: Kuup Pealtvaade | |
| Kontrollis | Erkki Jõgi | | |
| Kinnitas | Erkki Jõgi | | |
| EMÜ TS-TN | | Leht: 4/5 | Tähis: TN 17/130261 G 01 03 D |

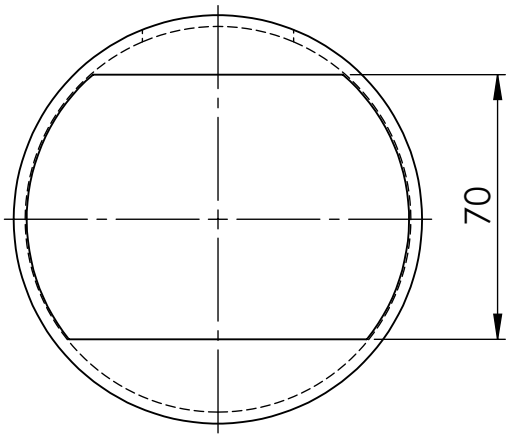



*Detaili pikkus l = 40mm*

| | | | | |
|---|---|---|---|---|
| | *Materjal:* *POM Acetal Copolymer* | *Näitamata piirhälbed:* *ISO 2768-m* | *Mass* *54 g* | *Mõõt* *2:1* |
| *Teostas* | *Kaljo Valk* | *Nimetus:* *Trummel* | | |
| *Kontrollis* | *Erkki Jõgi* | | | |
| *Kinnitas* | *Erkki Jõgi* | | | |
| *EMÜ TS-TN* | | *Leht:* *5/5* | *Tähis:* *TN 17/130261 G 01 04 D* | |

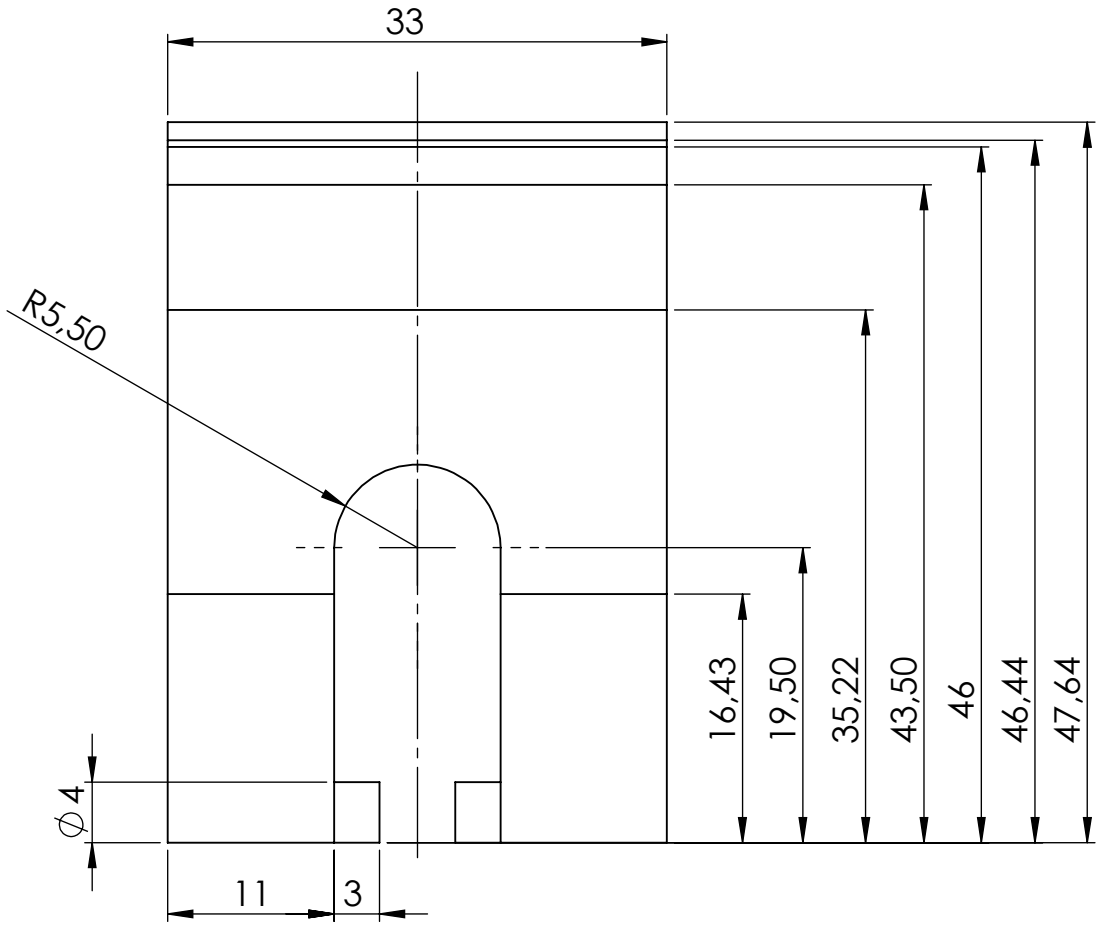
# LISA H



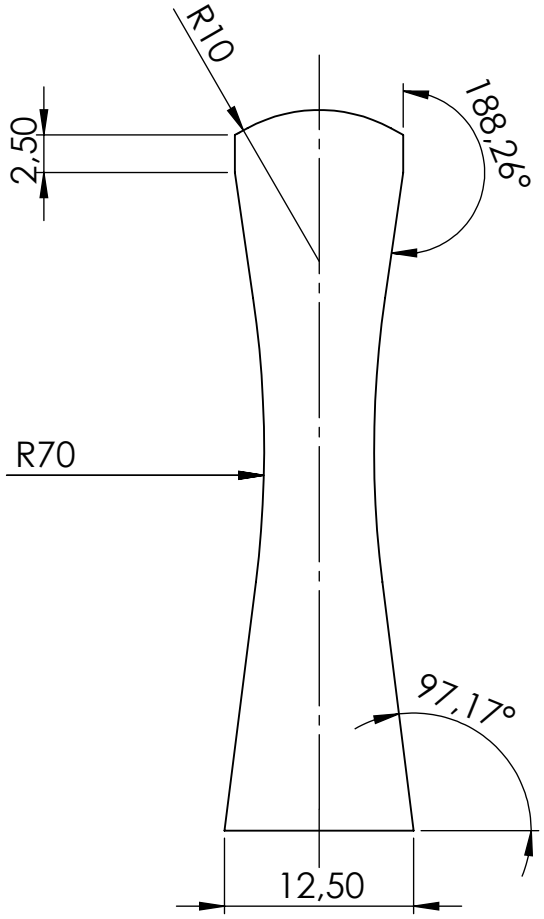




| | | | | |
|---|---|---|---|---|
| | Materjal: POM Acetal Copolymer | Näitamata piirhälbed: ISO 2768-m | Mass 857g | Mõõt 1:2 |
| Teostas | Kaljo Valk | Nimetus: Kaalumiskauss | | |
| Kontrollis | Erkki Jõgi | | | |
| Kinnitas | Erkki Jõgi | | | |
| EMÜ TS-TN | | Leht: 1/1 | Tähis: TN 17/130261 H 01 00 D | |

# LISA I



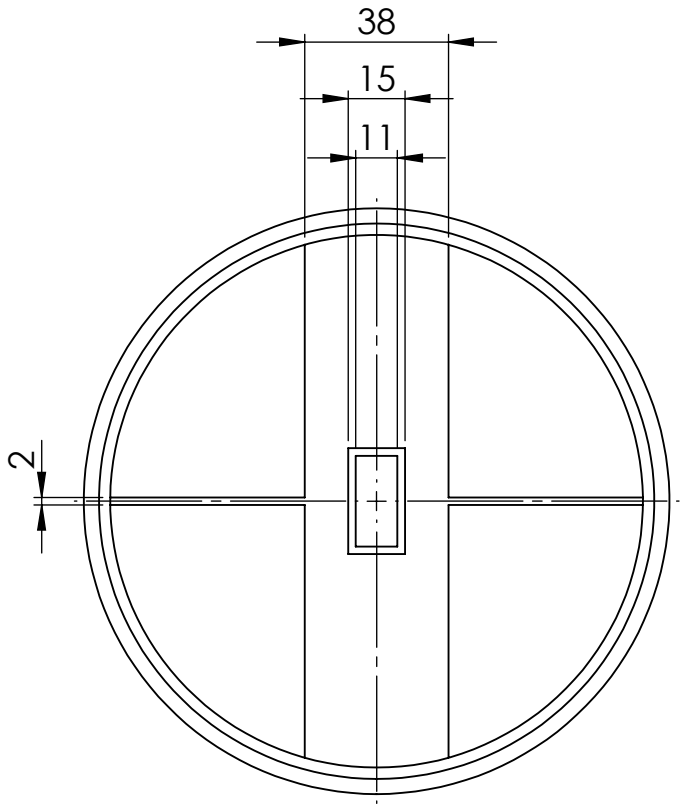

| | | | | |
|---|---|---|---|---|
| | Materjal: POM Acetal Copolymer | Näitamata piirhälbed: ISO 2768-m | Mass 17g | Mõõt 2:1 |
| Teostas | Kaljo Valk | Nimetus: Kaane käepide Eestvaade | | |
| Kontrollis | Erkki Jõgi | | | |
| Kinnitas | Erkki Jõgi | | | |
| EMÜ TS-TN | | Leht: 1/4 | Tähis: TN 17/130261 I 01 00 D | |

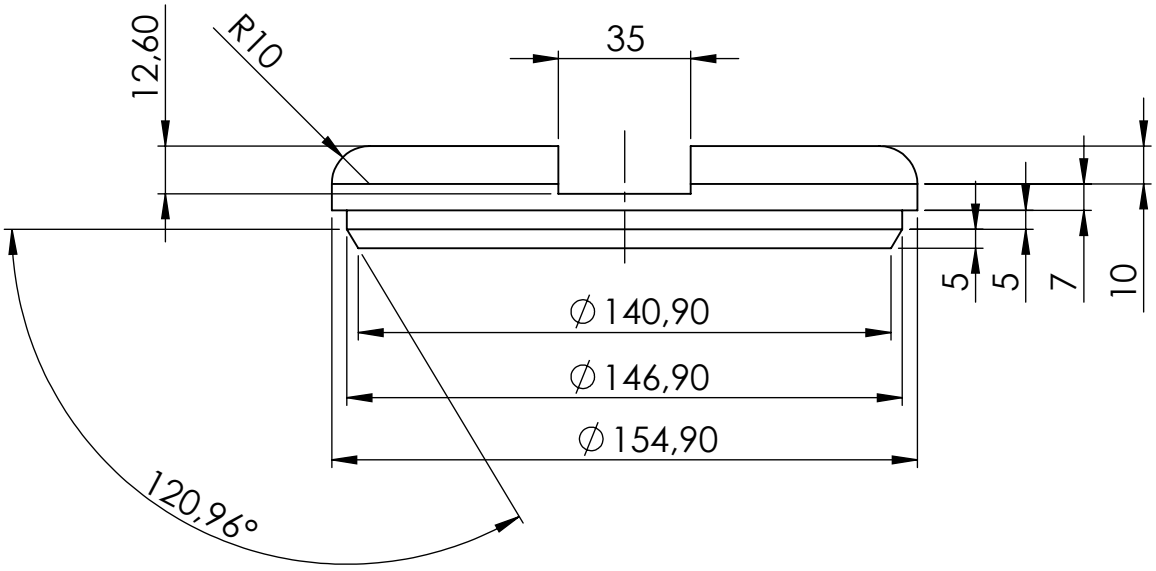



| | | | | |
|---|---|---|---|---|
| | Materjal: POM Acetal Copolymer | Näitamata piirhälbed: ISO 2768-m | Mass 17g | Mõõt 2:1 |

| | | |
|---|---|---|
| Teostas | Kaljo Valk | Nimetus: Kaane käepide Külgvaade |
| Kontrollis | Erkki Jõgi | |
| Kinnitas | Erkki Jõgi | |

| | | |
|---|---|---|
| EMÜ TS-TN | Leht: 2/4 | Tähis: TN 17/130261 I 01 01 D |

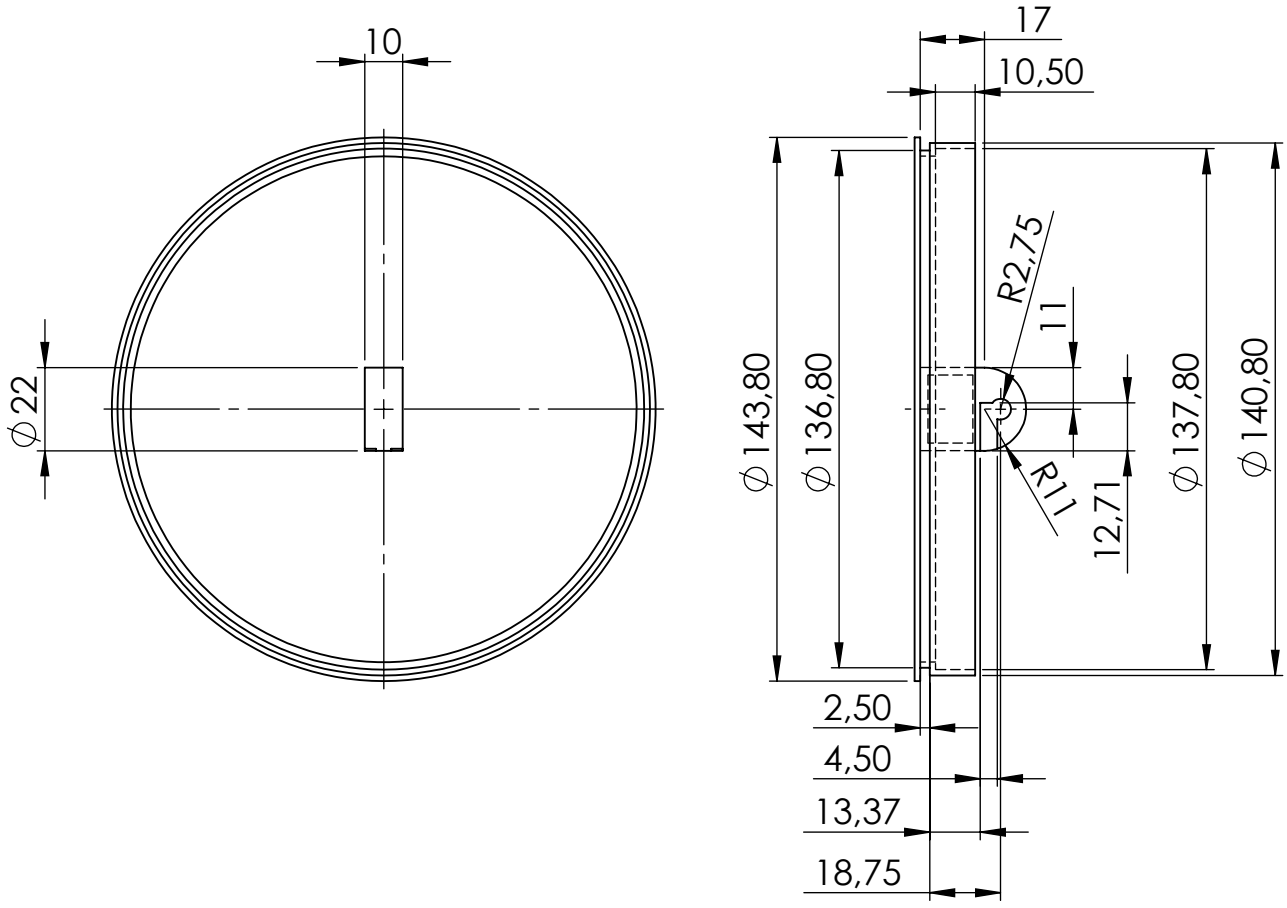



| | | | | |
|---|---|---|---|---|
| | Materjal: POM Acetal Copolymer | Näitamata piirhälbed: ISO 2768-m | Mass 100g | Mõõt 1:2 |

| | | |
|---|---|---|
| Teostas | Kaljo Valk | Nimetus: Kaas Ülemine osa |
| Kontrollis | Erkki Jõgi | |
| Kinnitas | Erkki Jõgi | |

| | | |
|---|---|---|
| EMÜ TS-TN | Leht: 3/4 | Tähis: TN 17/130261 I 01 02 D |

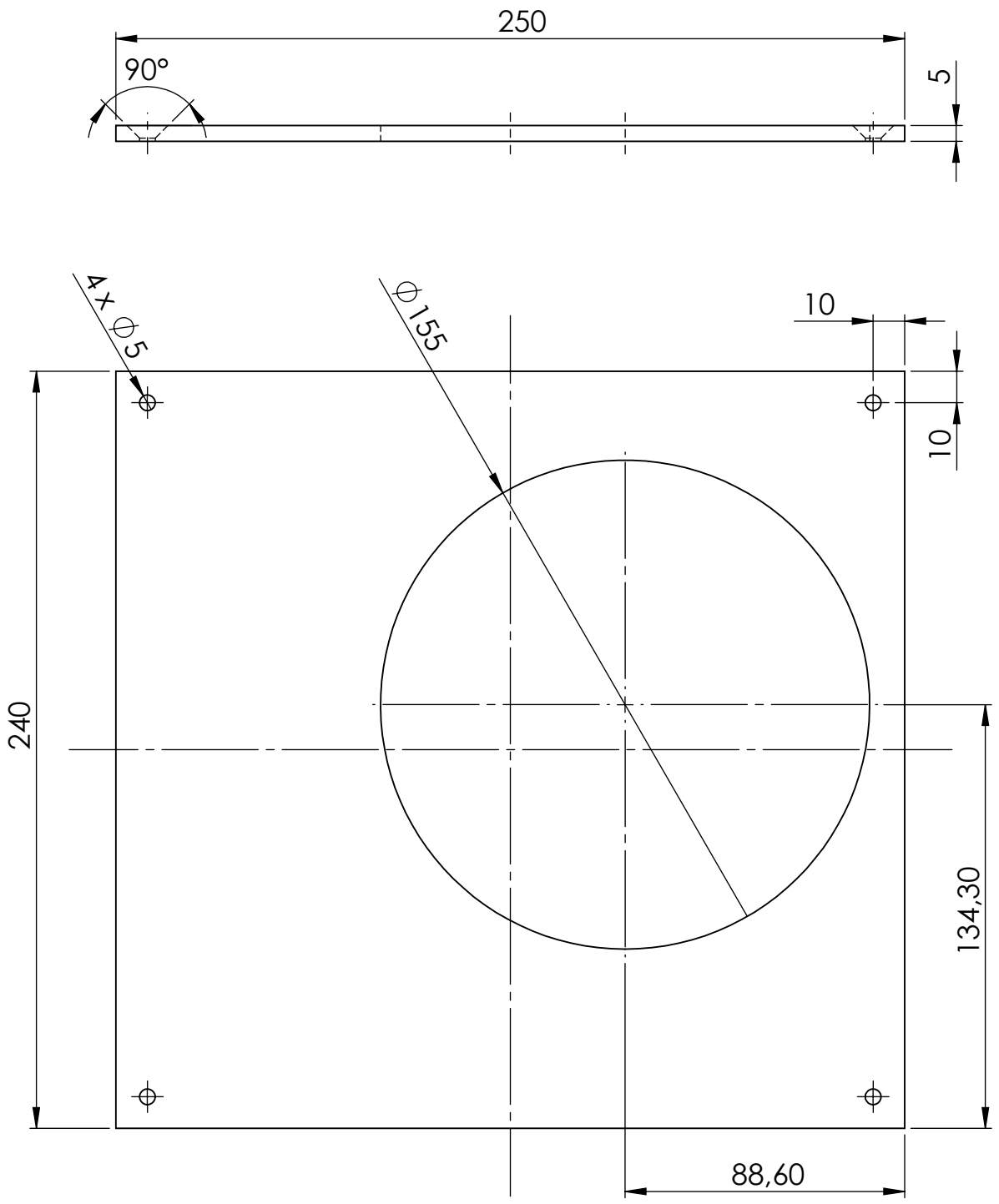



| | | | | |
|---|---|---|---|---|
| | Materjal: POM Acetal Copolymer | Näitamata piirhälbed: ISO 2768-m | Mass 56g | Mõõt 1:2 |

| | | |
|---|---|---|
| Teostas | Kaljo Valk | Nimetus: Kaas Alumine osa |
| Kontrollis | Erkki Jõgi | |
| Kinnitas | Erkki Jõgi | |

| | | |
|---|---|---|
| EMÜ TS-TN | Leht: 4/4 | Tähis: TN 17/130261 I 01 03 D |

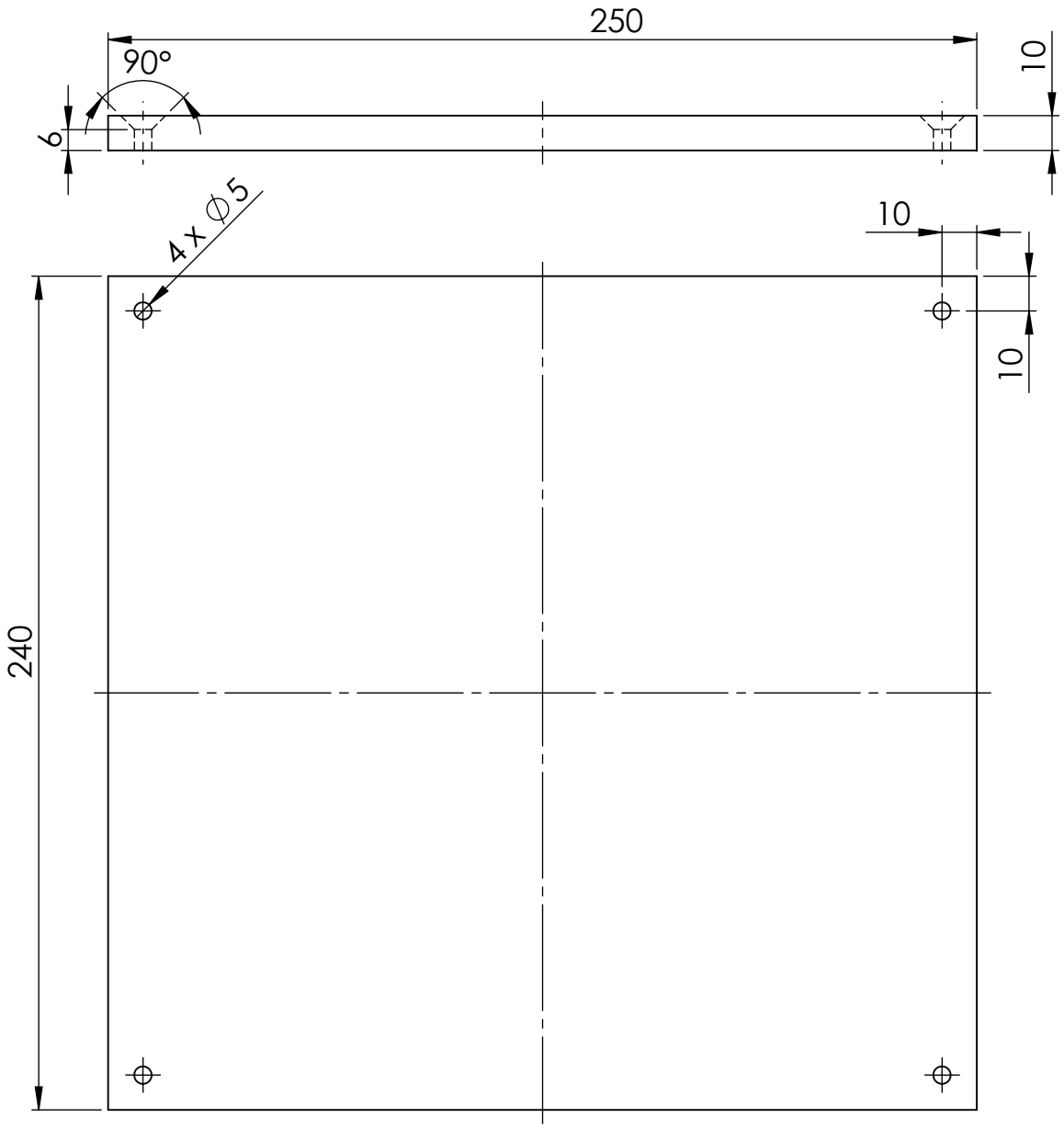
**LISA J**




| | | | | |
|---|---|---|---|---|
| | Materjal: Vineer | Näitamata piirhälbed: ISO 2768-m | Mass 114g | Mõõt 1:2 |

| | | |
|---|---|---|
| Teostas | Kaljo Valk | Nimetus: Korpus Ülemine pool |
| Kontrollis | Erkki Jõgi | |
| Kinnitas | Erkki Jõgi | |

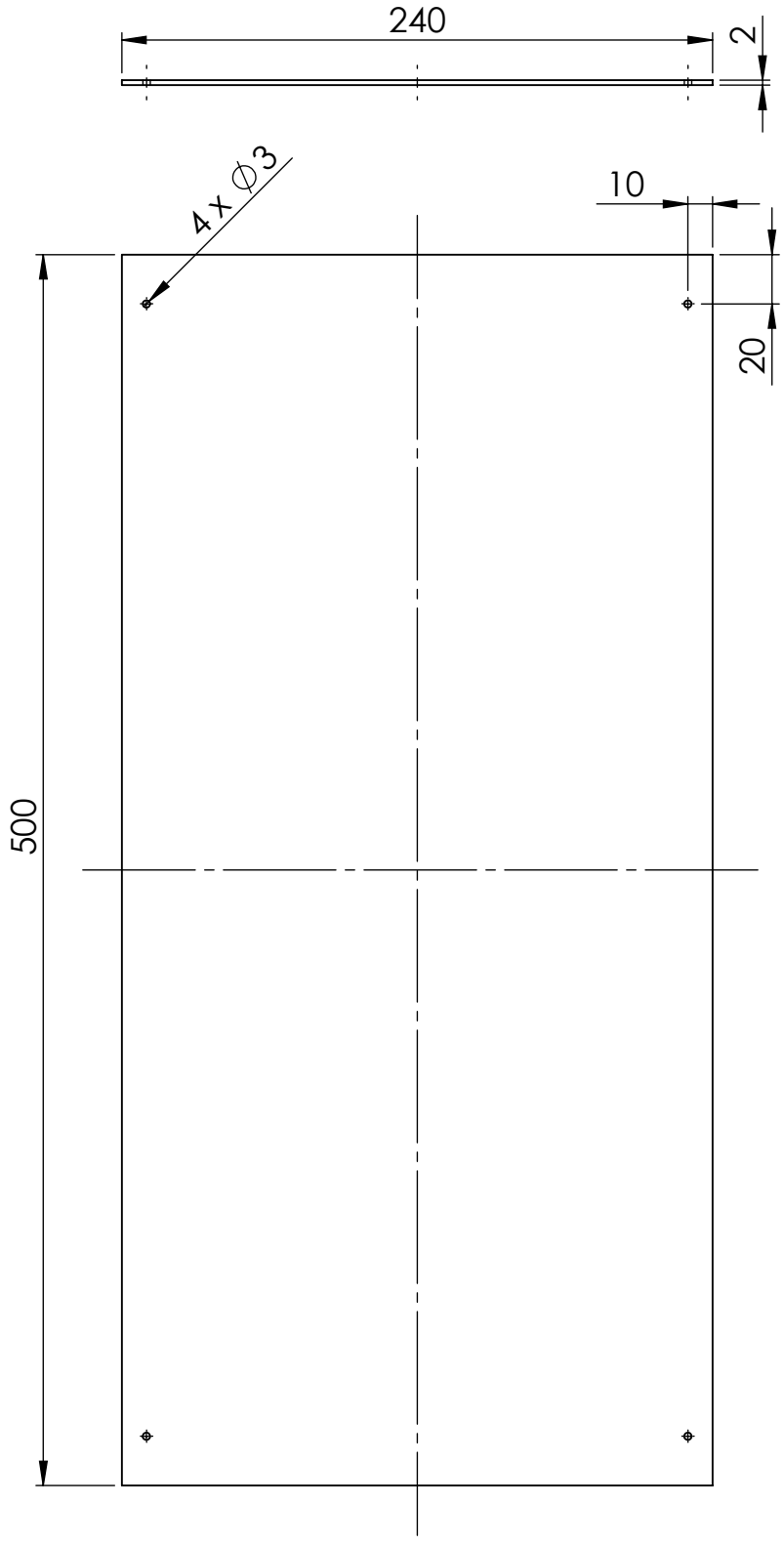
| | | |
|---|---|---|
| EMÜ TS-TN | Leht: 1/6 | Tähis: TN 17/130261 J 01 00 D |



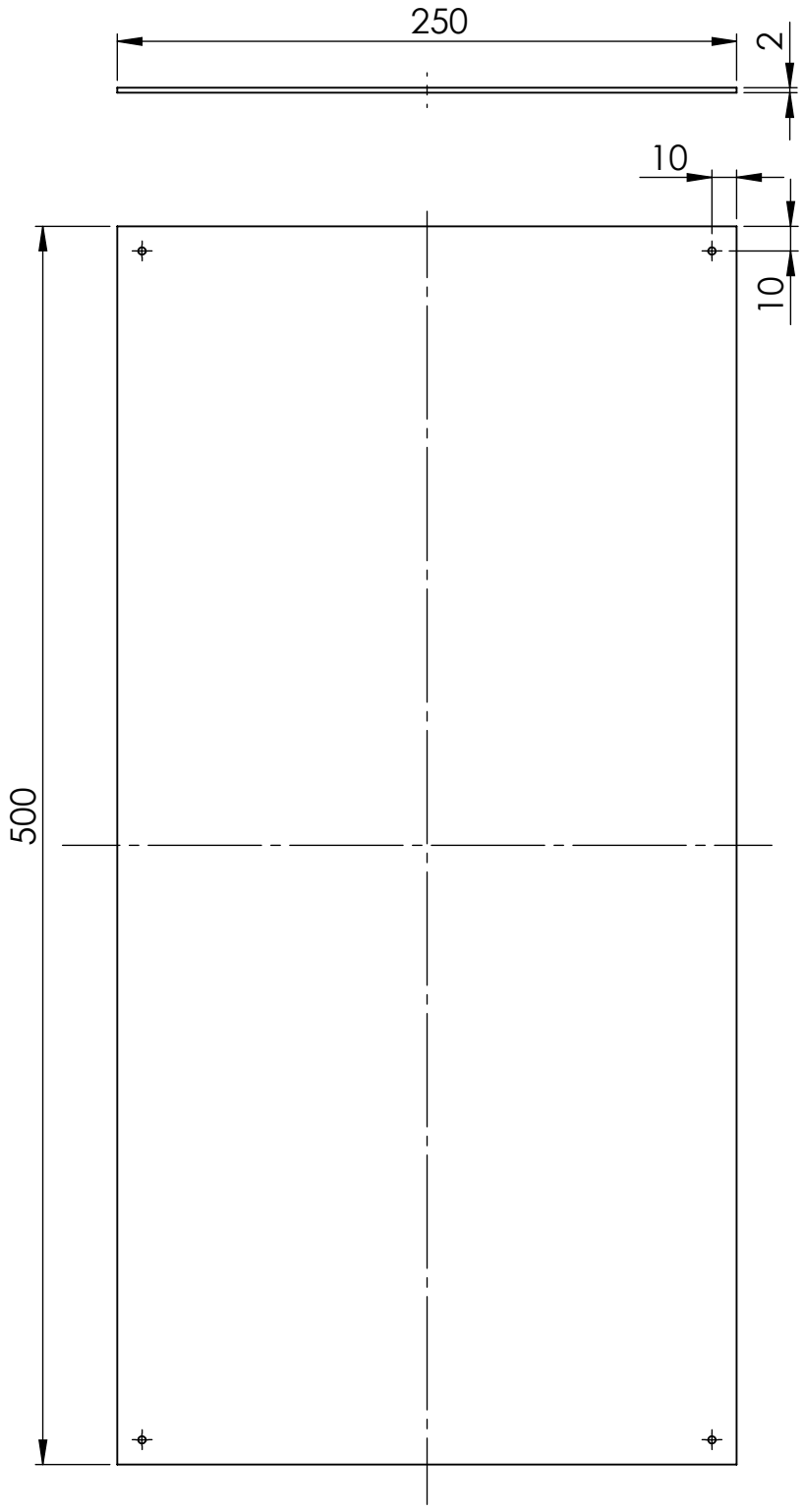

| | | | | |
|---|---|---|---|---|
| | Materjal: Vineer | Näitamata piirhälbed: ISO 2768-m | Mass 335g | Mõõt 1:2 |

| | | |
|---|---|---|
| Teostas | Kaljo Valk | Nimetus: Korpus Alus |
| Kontrollis | Erkki Jõgi | |
| Kinnitas | Erkki Jõgi | |

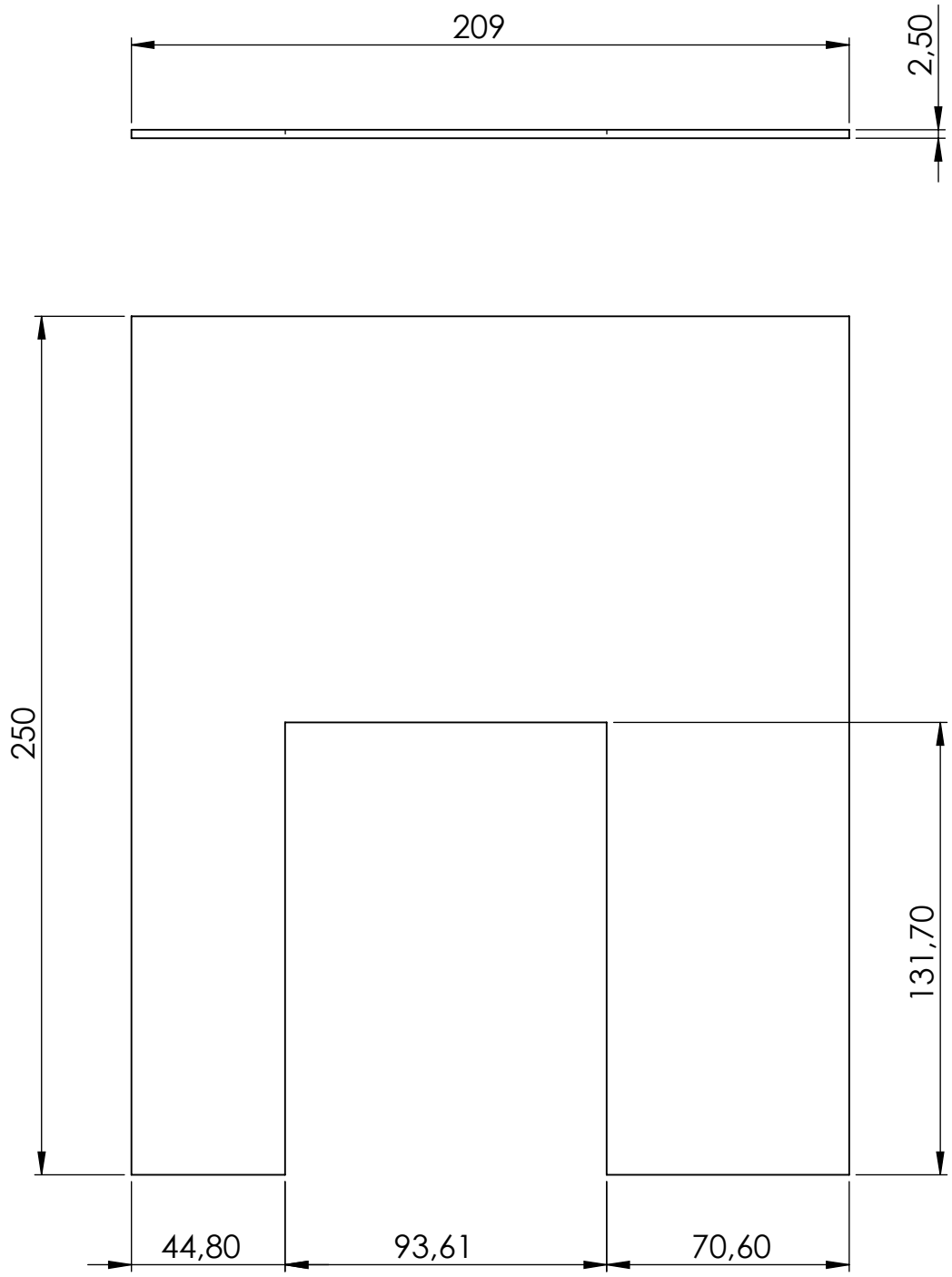
| | | |
|---|---|---|
| EMÜ TS-TN | Leht: 2/6 | Tähis: TN 17/130261 J 01 01 D |



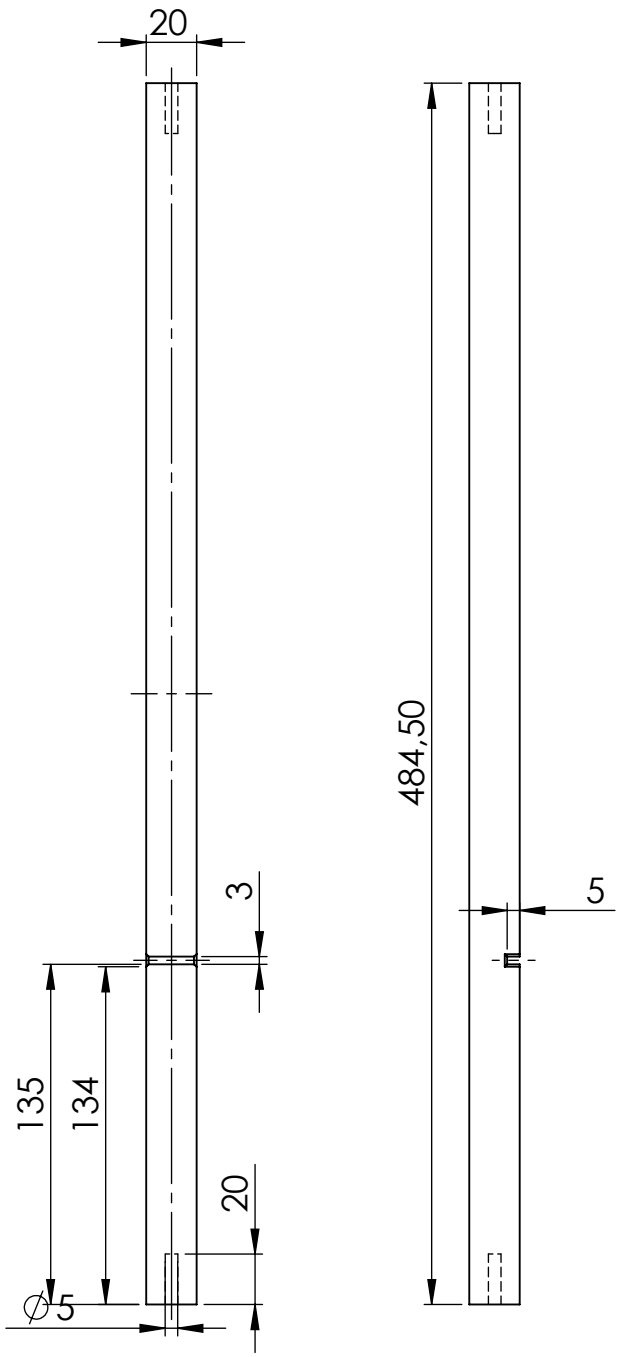

| | | | | |
|---|---|---|---|---|
| | Materjal: Vineer | Näitamata piirhälbed: ISO 2768-m | Mass 135g | Mõõt 1:2 |
| Teostas | Kaljo Valk | Nimetus: Korpus Küljepaneel 1 | | |
| Kontrollis | Erkki Jõgi | | | |
| Kinnitas | Erkki Jõgi | | | |
| EMÜ TS-TN | | Leht: 3/6 | Tähis: TN 17/130261 J 01 02 D | |




| | | | | |
|---|---|---|---|---|
| | Materjal: Vineer | Näitamata piirhälbed: ISO 2768-m | Mass 139g | Mõõt 1:3 |
| Teostas | Kaljo Valk | Nimetus: Korpus Küljepaneel 2 | | |
| Kontrollis | Erkki Jõgi | | | |
| Kinnitas | Erkki Jõgi | | | |
| EMÜ TS-TN | | Leht: 4/6 | Tähis: TN 17/130261 J 01 03 D | |




| | | | | |
|---|---|---|---|---|
| | Materjal: Vineer | Näitamata piirhälbed: ISO 2768-m | Mass 55g | Mõõt 1:2 |
| Teostas | Kaljo Valk | Nimetus: Korpus Vahelüli | | |
| Kontrollis | Erkki Jõgi | | | |
| Kinnitas | Erkki Jõgi | | | |
| EMÜ TS-TN | | Leht: 5/6 | Tähis: TN 17/130261 J 01 04 D | |




| | | | | |
|---|---|---|---|---|
| | Materjal: Vineer | Näitamata piirhälbed: ISO 2768-m | Mass 107g | Mõõt 1:3 |
| Teostas | Kaljo Valk | Nimetus: Korpus Küljtugi | | |
| Kontrollis | Erkki Jõgi | | | |
| Kinnitas | Erkki Jõgi | | | |
| EMÜ TS-TN | | Leht: 6/6 | Tähis: TN 17/130261 J 01 05 D | |

**LISA K**

LISA K Reaalajakella mooduli põhiprogrammi kood

```
#define F_CPU 2000000UL
#include <avr/io.h>
#include <util/delay.h>
#include <stdlib.h>
#include <string.h>
#include <avr/sleep.h>
#include <avr/power.h>
#include <avr/interrupt.h>
#include <avr/eeprom.h>
#include "UART_lib_v0.5_2Mhz.h"
#include "RTC_DS3231_I2C.v0.1.h"

int main(void){

	clock_prescale_set(clock_div_8);
	UART_init();
	I2C_Init();
	sei();

	DDRC = (1<<DDC6)|(1<<DDC7);
	PORTC = (1<<PORTC6);

	uint8_t Hour = 18;
	uint8_t Min  = 18;
	uint8_t Sec  = 00;

	DS3231_Write(DS3231_REG_HOUR, (0x3F & Hour));

	DS3231_Write(DS3231_REG_HOUR, Hour);
	DS3231_Write(DS3231_REG_MIN, Min);
	DS3231_Write(DS3231_REG_SEC, Sec);

	while(1)
	{
		uint8_t hour = DS3231_Read(DS3231_REG_HOUR);
		UART_senduint82Dec(BcdToUint_8(hour));
		UART_SendString(":");
		uint8_t min = DS3231_Read(DS3231_REG_MIN);
		UART_senduint82Dec(BcdToUint_8(min));
		UART_SendString(":");
		uint8_t sec = DS3231_Read(DS3231_REG_SEC);
		UART_senduint82Dec(BcdToUint_8(sec));
		UART_sendNewline();

		_delay_ms(1000);
	}
}
```

LISA K Reaalajakella mooduli UART 2 MHz suhtluse kood

```
#ifndef UART
#define UART

void UART_init() {

	UCSR1C = (1<<UCSZ11)|(1<<UCSZ10);
	UBRR1 = 12;
	UCSR1B = (1 <<RXCIE1)|(1 <<RXEN1)|(1<<TXEN1);
}

void UART_sendChar(char data) {

	while(!(UCSR1A & (1<<UDRE1))) {
	}
	UDR1=data;
}

void UART_sendUnsignedChar(unsigned char data) {

	while(!(UCSR1A & (1<<UDRE1))) {
	}
	UDR1=data;
}

void UART_SendString(char arr[]) {

	int i =0;
	while (arr[i] != 0x00)
	{
		UART_sendChar(arr[i]);
		i++;
	}

}

void UART_SendStringUnsignedChar(unsigned char arr[]) {

	int i =0;
	while (arr[i] != 0x00)
	{
		UART_sendChar(arr[i]);
		i++;
	}
}
```

LISA K Reaalajakella mooduli UART 2 MHz suhtluse kood (jätk)

```
void UART_senduint82Dec(char data) {

	int	radix = 10;
	char	buffer[10];
	ltoa(data,buffer,radix);
	UART_SendString(buffer);
}

void UART_sendNewline(){

	UART_sendChar('\n');
	UART_sendChar('\r');
}

void _UART_send_byte2(uint16_t byte){

	uint16_t temp = byte;
	for(int8_t i=9;i>=0;i--){
		temp = byte;
		if (temp & (1<<i)){
			UART_sendChar(49);
		}else{
			UART_sendChar(48);
		}
	}
}

void UART_send_byte(uint8_t byte){

	uint8_t temp = byte;

	for(int8_t i=7;i>=0;i--){
		temp = byte;
		if (temp & (1<<i)){
			UART_sendChar(49);
		}else{
			UART_sendChar(48);
		}
	}
}
```

LISA K Reaalajakella mooduli UART 2 MHz suhtluse kood (jätk)

```
void UART_send_2x_byte(uint16_t byte_2x){

	uint16_t temp = byte_2x;

	for(int8_t i=15;i>=0;i--){
		temp = byte_2x;
		if (temp & (1<<i)){
			UART_sendChar(49);
		}else{
			UART_sendChar(48);
		}
	}
}


#endif /* UART */
```

LISA K Reaalajakella mooduli $I^2C$ liidese suhtluse kood

```
#ifndef RTC_DS3231_I2C
#define RTC_DS3231_I2C

// used libraries:
// https://exploreembedded.com/wiki/AVR_C_Library
// htps://libstock.mikroe.com/projects/view/1329/ds3231-rtc-with-avr

uint8_t DS3231_REG_SEC  = 0x00;
uint8_t DS3231_REG_MIN  = 0x01;
uint8_t DS3231_REG_HOUR  = 0x02;

#define DS3231_Read_addr          0b11010001
#define DS3231_Write_addr         0b11010000

uint8_t BcdToUint_8(uint8_t Bdc)
{
        return Bdc - 6 * (Bdc >> 4);
}

uint8_t Uint8ToBcd(uint8_t val)
{
        return val + 6 * (val / 10);
}

void I2C_Init()
{
        TWSR=0x00;
        TWBR=0x0C;
        TWCR=0x04;
}

void I2C_Start()
{
        TWCR = ((1<<TWINT) | (1<<TWSTA) | (1<<TWEN));
        while (!(TWCR & (1<<TWINT)));
}

void I2C_Stop(void)
{
        TWCR = ((1<< TWINT) | (1<<TWEN) | (1<<TWSTO));
        _delay_us(10) ;
}
```

LISA K Reaalajakella mooduli I$^2$C liidese suhtluse kood (jätk)

```
void I2C_Write(uint8_t dat)
{
	TWDR = dat ;
	TWCR = ((1<< TWINT) | (1<<TWEN));
	while (!(TWCR & (1 <<TWINT)));
}

uint8_t I2C_Read(unsigned char ack)
{
	TWCR = ((1<< TWINT) | (1<<TWEN) | (ack<<TWEA));
	while ( !(TWCR & (1 <<TWINT)));

	return TWDR;
}

uint8_t DS3231_Read(uint8_t address)
{
	I2C_Start();
	I2C_Write(DS3231_Write_addr);
	I2C_Write(address);
	I2C_Start();
	I2C_Write(DS3231_Read_addr);
	uint8_t value = I2C_Read(0);
	I2C_Stop();

	return value;
}

void DS3231_Write(uint8_t address, uint8_t value)
{
	I2C_Start();
	I2C_Write(DS3231_Write_addr);
	I2C_Write(address);
	I2C_Write(Uint8ToBcd(value));
	I2C_Stop();
}

#endif /* RTC_DS3231_I2C.h */
```

**LISA L**

LISA L arendusplaadi ja Raspberry Pi kommunikatsiooni kood

Arendusplaadi osa põhiprogramm:

```
#define F_CPU 2000000UL
#include <avr/io.h>
#include <util/delay.h>
#include <stdlib.h>
#include <string.h>
#include <avr/sleep.h>
#include <avr/power.h>
#include <avr/interrupt.h>
#include <avr/eeprom.h>
#include "UART_lib_v0.5_2Mhz.h"
#include "Loputoo_v.0.8_func.h"
#include "RTC_DS3231_I2C.v0.1.h"

int main(void){

	clock_prescale_set(clock_div_8);
	UART_init();
	I2C_Init();
	sei();

	DDRC = (1<<DDC6)|(1<<DDC7);
	PORTC = (1<<PORTC6);

	while(1){
		_delay_ms(500);
	}
}
```

LISA L arendusplaadi ja *Raspberry Pi* kommunikatsiooni kood. Arendusplaadi osa alamprogramm, katkestuste käsitlemine ning kommunikatsioon *RASPBERRY PI-ga*. UART suhtluseks kasutas autor LISAS K reaalajakella mooduli UART 2 MHz suhtluse koodi.

```
#ifndef SUPPORTIVE_FUNCTIONS
#define SUPPORTIVE_FUNCTIONS

#define true 1
#define false 0

uint8_t controlbyte EEMEM;
uint8_t dailyIntake EEMEM;
uint8_t kordaP2evas EEMEM;
uint8_t kellaaeg1 EEMEM;
uint8_t kellaaeg2 EEMEM;
uint8_t kellaaeg3 EEMEM;
uint8_t kellaaeg4 EEMEM;
uint8_t kellaaeg5 EEMEM;

uint8_t controlbyte;
uint8_t dailyIntake;
uint8_t kordaP2evas;
uint8_t kellaaeg1;
uint8_t kellaaeg2;
uint8_t kellaaeg3;
uint8_t kellaaeg4;
uint8_t kellaaeg5;

char RXstring[30];
char buffer[10];
char    previousChar = 'P';
uint8_t count = 0;
int             radix = 10;
int             stateStart = 0;

void sendConfiViaUART();
void makestring(char);
void EEPROMwriteSetting(char[]);
void emptystring();

ISR(USART1_RX_vect){
        char data = UDR1;
        if (data == 'x') {      sendConfiViaUART(); }

        if (data == 'a'){
                stateStart = 1;
                makestring(data);
        }else if (stateStart == 1 && data != 'h'){
```

```
                makestring(data);
        }else if (data == 'h'){
                makestring(data);
                stateStart = 2;
        }

        if (stateStart == 2){
                count = 0;
                EEPROMwriteSetting(RXstring);
                stateStart = 0;
        }
}


uint8_t currentchar = 'p';

void EEPROMwriteSetting(char RXstring[]){

        uint8_t debug = false;

        char dailyIntakeCharArr[3] = {0,0,0};
        char kordaP2evasCharArr[3] = {0,0,0};
        char kellaaeg1CharArr[3] = {0,0,0};
        char kellaaeg2CharArr[3] = {0,0,0};
        char kellaaeg3CharArr[3] = {0,0,0};
        char kellaaeg4CharArr[3] = {0,0,0};
        char kellaaeg5CharArr[3] = {0,0,0};

        for(int8_t i=0;i<25;i++){
                if (debug == true){
                        UART_SendString("RXstring[");
                        UART_senduint82Dec(i);
                        UART_SendString("] = ");
                        UART_sendChar(RXstring[i]);
                        UART_sendNewline();
                }
                if (RXstring[i] == 'a'){
                        currentchar = 'a';

                        if (debug == true){
                                UART_SendString("currentchar = ");
                                UART_sendChar(currentchar);
                                UART_sendNewline();
                        }
                }
                else if (RXstring[i] != 'a' && currentchar == 'a' && RXstring[i] != 'b')
                {
                        dailyIntakeCharArr[count] = RXstring[i];
```

```
		count++;
	}
	if (RXstring[i] == 'b')
	{
		currentchar = 'b';
		count = 0;

		if (debug == true){
			UART_sendNewline();
			UART_SendString("currentchar = ");
			UART_sendChar(currentchar);
			UART_sendNewline();
		}
	}
	else if (RXstring[i] != 'b' && currentchar == 'b' && RXstring[i] != 'c')
	{
		kordaP2evasCharArr[count] = RXstring[i];
		count++;
	}
	//-----------------------------------------------------------------
	if (RXstring[i] == 'c')
	{
		currentchar = 'c';
		count = 0;

		if (debug == true){
			UART_sendNewline();
			UART_SendString("currentchar = ");
			UART_sendChar(currentchar);
			UART_sendNewline();
		}
	}
	else if (RXstring[i] != 'c' && currentchar == 'c' && RXstring[i] != 'd')
	{
		kellaaeg1CharArr[count] = RXstring[i];
		count++;
	}
	//-----------------------------------------------------------------
	if (RXstring[i] == 'd')
	{
		currentchar = 'd';
		count = 0;

		if (debug == true){
			UART_sendNewline();
			UART_SendString("currentchar = ");
			UART_sendChar(currentchar);
			UART_sendNewline();
```

```
		}
	}
	else if (RXstring[i] != 'd' && currentchar == 'd' && RXstring[i] != 'e')
	{
		kellaaeg2CharArr[count] = RXstring[i];
		count++;
	}

	//-----------------------------------------------------------------

	if (RXstring[i] == 'e')
	{
		currentchar = 'e';
		count = 0;

		if (debug == true){
			UART_sendNewline();
			UART_SendString("currentchar = ");
			UART_sendChar(currentchar);
			UART_sendNewline();
		}
	}
	else if (RXstring[i] != 'e' && currentchar == 'e' && RXstring[i] != 'f')
	{
		kellaaeg3CharArr[count] = RXstring[i];
		count++;
	}
	//-----------------------------------------------------------------

	if (RXstring[i] == 'f')
	{
		currentchar = 'f';
		count = 0;

		if (debug == true){
			UART_sendNewline();
			UART_SendString("currentchar = ");
			UART_sendChar(currentchar);
			UART_sendNewline();
		}
	}
	else if (RXstring[i] != 'f' && currentchar == 'f' && RXstring[i] != 'g')
	{
		kellaaeg4CharArr[count] = RXstring[i];
		count++;
	}
	//-----------------------------------------------------------------
```

```
		if (RXstring[i] == 'g')
		{
			currentchar = 'g';
			count = 0;

			if (debug == true){
				UART_sendNewline();
				UART_SendString("currentchar = ");
				UART_sendChar(currentchar);
				UART_sendNewline();
			}
		}
		else if (RXstring[i] != 'g' && currentchar == 'g' && RXstring[i] != 'h')
		{
			kellaaeg5CharArr[count] = RXstring[i];
			count++;
		}
		//---------------------------------------------------------------------

		if (RXstring[i] == 'h')
		{
			currentchar = 'h';
			count = 0;
			break;
		}
	}

	uint8_t dailyIntakeResoult = atoi(dailyIntakeCharArr);
	uint8_t kordaP2evasResoult   = atoi(kordaP2evasCharArr);
	uint8_t kellaaeg1Resoult     = atoi(kellaaeg1CharArr);
	uint8_t kellaaeg2Resoult     = atoi(kellaaeg2CharArr);
	uint8_t kellaaeg3Resoult     = atoi(kellaaeg3CharArr);
	uint8_t kellaaeg4Resoult     = atoi(kellaaeg4CharArr);
	uint8_t kellaaeg5Resoult     = atoi(kellaaeg5CharArr);

	eeprom_write_byte(&dailyIntake, dailyIntakeResoult);
	eeprom_write_byte(&kordaP2evas, kordaP2evasResoult);
	eeprom_write_byte(&kellaaeg1, kellaaeg1Resoult);
	eeprom_write_byte(&kellaaeg2, kellaaeg2Resoult);
	eeprom_write_byte(&kellaaeg3, kellaaeg3Resoult);
	eeprom_write_byte(&kellaaeg4, kellaaeg4Resoult);
	eeprom_write_byte(&kellaaeg5, kellaaeg5Resoult);

	if (debug == true){
		UART_SendString("EEread_dailyIntake:");
		UART_senduint82Dec(eeprom_read_byte(&dailyIntake));
		UART_sendNewline();
		UART_SendString("EEread_kellaaeg1:");
```

```
                UART_senduint82Dec(eeprom_read_byte(&kellaaeg1));
                UART_sendNewline();
                UART_SendString("EEread_kellaaeg2:");
                UART_senduint82Dec(eeprom_read_byte(&kellaaeg2));
                UART_sendNewline();
                UART_SendString("EEread_kellaaeg3:");
                UART_senduint82Dec(eeprom_read_byte(&kellaaeg3));
                UART_sendNewline();
                UART_SendString("EEread_kellaaeg4:");
                UART_senduint82Dec(eeprom_read_byte(&kellaaeg4));
                UART_sendNewline();
                UART_SendString("EEread_kellaaeg5:");
                UART_senduint82Dec(eeprom_read_byte(&kellaaeg5));
                UART_sendNewline();

        }

        if (debug == true){
                UART_SendString("dailyIntakeCharArr:");
                UART_SendString(dailyIntakeCharArr);
                UART_sendNewline();
                UART_SendString("kordaP2evasCharArr:");
                UART_SendString(kordaP2evasCharArr);
                UART_sendNewline();
                UART_SendString("kellaaeg1CharArr:");
                UART_SendString(kellaaeg1CharArr);
                UART_sendNewline();
                UART_SendString("kellaaeg2CharArr:");
                UART_SendString(kellaaeg2CharArr);
                UART_sendNewline();
                UART_SendString("kellaaeg3CharArr:");
                UART_SendString(kellaaeg3CharArr);
                UART_sendNewline();
                UART_SendString("kellaaeg4CharArr:");
                UART_SendString(kellaaeg4CharArr);
                UART_sendNewline();
                UART_SendString("kellaaeg5CharArr:");
                UART_SendString(kellaaeg5CharArr);
                UART_sendNewline();
        }
}

void makestring(char data) {
        RXstring[count] = data;
        count++;
}

void emptystring() {
```

```
	RXstring[0] = 0;
	RXstring[1] = 0;
	count = 0;
}

void sendConfiViaUART() {
	_delay_ms(200);

	UART_SendString("a");

	UART_senduint82Dec(eeprom_read_byte(&dailyIntake));

	UART_SendString("b");
	UART_senduint82Dec(eeprom_read_byte(&kordaP2evas));

	UART_SendString("c");
	UART_senduint82Dec(eeprom_read_byte(&kellaaeg1));

	UART_SendString("d");
	UART_senduint82Dec(eeprom_read_byte(&kellaaeg2));

	UART_SendString("e");
	UART_senduint82Dec(eeprom_read_byte(&kellaaeg3));

	UART_SendString("f");
	UART_senduint82Dec(eeprom_read_byte(&kellaaeg4));

	UART_SendString("g");
	UART_senduint82Dec(eeprom_read_byte(&kellaaeg5));

	UART_SendString("h");
	UART_sendNewline();
}

void enterInfiniteLoop(uint8_t n) {
	DDRC = (1<<DDC7);

	while(1){
		for (int i=0; i < n; i++){
			PORTC = (1<<PORTC7);
			_delay_ms(500);
			PORTC = (0<<PORTC7);
		}
		_delay_ms(1000);
	}
}

uint8_t uinreadControlByteFromEEPROM() {
```

```
	uint8_t  controlbyte;
	controlbyte = eeprom_read_byte(&controlbyte);
	UART_send_byte(controlbyte);

	return controlbyte;
}
void writeDefaultSettingsToEEPROM() {

	int		radix = 10;
	char	buffer[10];

	ltoa(eeprom_read_byte(&dailyIntake),buffer,radix);
	UART_SendString(buffer);
	UART_sendNewline();
	ltoa(eeprom_read_byte(&kordaP2evas),buffer,radix);
	UART_SendString(buffer);
	UART_sendNewline();
	ltoa(eeprom_read_byte(&kellaaeg1),buffer,radix);
	UART_SendString(buffer);
	UART_sendNewline();
	ltoa(eeprom_read_byte(&kellaaeg2),buffer,radix);
	UART_SendString(buffer);
	UART_sendNewline();
	ltoa(eeprom_read_byte(&kellaaeg3),buffer,radix);
	UART_SendString(buffer);
	UART_sendNewline();
	ltoa(eeprom_read_byte(&kellaaeg4),buffer,radix);
	UART_SendString(buffer);
	UART_sendNewline();
	ltoa(eeprom_read_byte(&kellaaeg5),buffer,radix);
	UART_SendString(buffer);
	UART_sendNewline();

}

#endif /* SUPPORTIVE_FUNCTIONS */
```

LISA L arendusplaadi ja *Raspberry Pi* kommunikatsiooni kood. *Raspberry Pi* kommunikatsiooni ja veebiserveri osa:

```
import RPi.GPIO as GPIO #Sisend/väljund viikude paketi lisamine
import time, sys, serial

from flask import Flask, render_template, flash, request
from wtforms import TextField, TextAreaField, StringField, SubmitField, Form, validators

GPIO.setwarnings(False)
GPIO.setmode(GPIO.BOARD)
GPIO.setup(7, GPIO.OUT)
GPIO.output(7, True)
print ("GPIO.output(7, True)")

class Settings:

    dailyIntake = 0
    kordaP2evas = 0
    kellaaeg1   = 0
    kellaaeg2   = 0
    kellaaeg3   = 0
    kellaaeg4   = 0
    kellaaeg5   = 0
    kellaajad   = ""


    #constructors
    def __init__(self):
        self.dailyIntake = Settings.dailyIntake
        self.kordaP2evas = Settings.kordaP2evas

        self.kellaaeg1 = Settings.kellaaeg1
        self.kellaaeg2 = Settings.kellaaeg2
        self.kellaaeg3 = Settings.kellaaeg3
        self.kellaaeg4 = Settings.kellaaeg4
        self.kellaaeg5 = Settings.kellaaeg5

        self.kellaajad = Settings.kellaajad

    def getdailyIntake(self):
        return Settings.dailyIntake
    def setdailyIntake(self, newValue):
        Settings.dailyIntake = newValue

    def getkordaP2evas(self):
        return Settings.kordaP2evas
    def setkordaP2evas(self, newValue):
        Settings.kordaP2evas = newValue
```

LISA L arendusplaadi ja *Raspberry Pi* kommunikatsiooni kood. *Raspberry Pi* kommunikatsiooni ja veebiserveri osa (jätk)

```
    def getkellaaeg1(self):
        return Settings.kellaaeg1
    def setkellaaeg1(self, newValue):
        Settings.kellaaeg1 = newValue

    def getkellaaeg2(self):
        return Settings.kellaaeg2
    def setkellaaeg2(self, newValue):
        Settings.kellaaeg2 = newValue

    def getkellaaeg3(self):
        return Settings.kellaaeg3
    def setkellaaeg3(self, newValue):
        Settings.kellaaeg3 = newValue

    def getkellaaeg4(self):
        return Settings.kellaaeg4
    def setkellaaeg4(self, newValue):
        Settings.kellaaeg4 = newValue

    def getkellaaeg5(self):
        return Settings.kellaaeg5
    def setkellaaeg5(self, newValue):
        Settings.kellaaeg5 = newValue

    def getkellaajad(self):
        return Settings.kellaajad
    def setkellaajad(self, newValue):
        Settings.kellaajad = newValue


def writeSettingstoVariables(strIn):
    print("writeSettingstoVariables() START")

    strIn = strIn.strip('a')
    print(strIn)

    dailyIntake = int(strIn.split("b")[0])

    kordaP2evas = strIn.split("b")[1]
    kordaP2evas = kordaP2evas.split("c")[0]

    kellaaeg1 = strIn.split("c")[1]
    kellaaeg1 = kellaaeg1.split("d")[0]

    kellaaeg2 = strIn.split("d")[1]
    kellaaeg2 = kellaaeg2.split("e")[0]
```

LISA L arendusplaadi ja *Raspberry Pi* kommunikatsiooni kood. *Raspberry Pi* kommunikatsiooni ja veebiserveri osa (jätk)

```
    kellaaeg3 = strIn.split("e")[1]
    kellaaeg3 = kellaaeg3.split("f")[0]

    kellaaeg4 = strIn.split("f")[1]
    kellaaeg4 = kellaaeg4.split("g")[0]

    kellaaeg5 = strIn.split("g")[1].strip('h')

    obj1 = Settings()
    obj1.setdailyIntake(dailyIntake)
    obj1.setkordaP2evas(kordaP2evas)
    obj1.setkellaaeg1(kellaaeg1)
    obj1.setkellaaeg2(kellaaeg2)
    obj1.setkellaaeg3(kellaaeg3)
    obj1.setkellaaeg4(kellaaeg4)
    obj1.setkellaaeg5(kellaaeg5)

    if kordaP2evas == "1":
        obj1.setkellaajad(kellaaeg1)

    if kordaP2evas == "2":
        kellaajad = kellaaeg1 + "," + kellaaeg2
        obj1.setkellaajad(kellaajad)

    if kordaP2evas == "3":
        kellaajad = kellaaeg1 + "," + kellaaeg2 + "," + kellaaeg3
        obj1.setkellaajad(kellaajad)

    if kordaP2evas == "4":
        kellaajad = kellaaeg1 + "," + kellaaeg2 + "," + kellaaeg3 + "," + kellaaeg4
        obj1.setkellaajad(kellaajad)


    if kordaP2evas == "5":
        kellaajad = kellaaeg1 + "," + kellaaeg2 + "," + kellaaeg3 + "," + kellaaeg4 + "," +
kellaaeg5
        obj1.setkellaajad(kellaajad)


    print(obj1.getdailyIntake())
    print(obj1.getkordaP2evas())
    print(obj1.getkellaaeg1())
    print(obj1.getkellaaeg2())
    print(obj1.getkellaaeg3())
    print(obj1.getkellaaeg4())
    print(obj1.getkellaaeg5())
    print(obj1.getkellaajad())
```

LISA L arendusplaadi ja *Raspberry Pi* kommunikatsiooni kood. *Raspberry Pi* kommunikatsiooni ja veebiserveri osa (jätk)

```
def retrieveconfigFromAtmega():

    ser = serial.Serial(
    port='/dev/ttyUSB0',
    baudrate=9600,
    parity=serial.PARITY_NONE,
    stopbits=serial.STOPBITS_ONE,
    bytesize=serial.EIGHTBITS,
    timeout=0
    )

    if ser.isOpen():
        ser.flushInput() #flush input buffer, discarding all its contents
        ser.flushOutput()#flush output buffer, aborting current output
                    #and discard all that is in buffer

        ser.write(b'x')  #ask for settings
        time.sleep(0.5)  #give the serial port sometime to receive the data
        s = ser.read(30).decode().strip('\r\n')   # read up to ten bytes (timeout)

        if s != '':
            #print(s)
            #print("s != "")
            writeSettingstoVariables(s)
    ser.close()

def sendconfigToAtmega():

    ser = serial.Serial(
    port='/dev/ttyUSB0',
    baudrate=9600,
    parity=serial.PARITY_NONE,
    stopbits=serial.STOPBITS_ONE,
    bytesize=serial.EIGHTBITS,
    timeout=0
    )

    obj1 = Settings()

    if ser.isOpen():
        ser.flushInput() #flush input buffer, discarding all its contents
        ser.flushOutput()#flush output buffer, aborting current output
                    #and discard all that is in buffer
        print(obj1.getdailyIntake())
        print(obj1.getkordaP2evas())
        print(obj1.getkellaaeg1())
        print(obj1.getkellaaeg2())
```

LISA L arendusplaadi ja *Raspberry Pi* kommunikatsiooni kood. *Raspberry Pi* kommunikatsiooni ja veebiserveri osa (jätk)

```
            print(obj1.getkellaaeg3())
            print(obj1.getkellaaeg4())
            print(obj1.getkellaaeg5())
            print(obj1.getkellaajad())
            outputStr = "a" + str(obj1.getdailyIntake()) + "b" + str(obj1.getkordaP2evas()) +
"c" + str(obj1.getkellaaeg1()) + "d" + str(obj1.getkellaaeg2()) + "e" +
str(obj1.getkellaaeg3()) + "f" + str(obj1.getkellaaeg4()) + "g" + str(obj1.getkellaaeg5()) +
"h"

            ser.write(outputStr.encode())  #send settings

        ser.close()

retrieveconfigFromAtmega()

DEBUG = True
app = Flask(__name__)
app.config.from_object(__name__)
app.config['SECRET_KEY'] = '7d441f2746365467d441f2b6176a'

class ReusableForm(Form):
    dailyIntake = TextField('', validators=[validators.required(), validators.Length(min=1,
max=2)])
    kordaP2evas = TextField('', validators=[validators.required(), validators.Length(min=1,
max=1)])
    kellaaegadel= TextField('', validators=[validators.required()]) #24h format

@app.route("/", methods=['GET', 'POST'])
def hello():

    form = ReusableForm(request.form)

    obj1 = Settings()
    var_dailyIntake = obj1.getdailyIntake()
    var_kordaP2evas = obj1.getkordaP2evas()

    var_kellaaeg1= obj1.getkellaaeg1()
    var_kellaaeg2= obj1.getkellaaeg2()
    var_kellaaeg3= obj1.getkellaaeg3()
    var_kellaaeg4= obj1.getkellaaeg4()
    var_kellaaeg5= obj1.getkellaaeg5()
    var_kellaajad=obj1.getkellaajad()

    if request.method == 'POST':

        if form.validate():
            dailyIntake=request.form['dailyIntake']
```

LISA L arendusplaadi ja *Raspberry Pi* kommunikatsiooni kood. *Raspberry Pi* kommunikatsiooni ja veebiserveri osa (jätk)

```
        kordaP2evas=request.form['kordaP2evas']
        kellaaegadel=request.form['kellaaegadel']

        obj1.setdailyIntake(dailyIntake)
        obj1.setkordaP2evas(kordaP2evas)
        obj1.setkellaajad(kellaaegadel.replace(".", ","))

        print ("----------------------------------")
        print ("dailyIntake: ", dailyIntake, " g ")
        print ("kordaP2evas: ", kordaP2evas)
        print ("kellaaegadel: ", kellaaegadel)
        print ("----------------------------------")
        sendconfigToAtmega()
        retrieveconfigFromAtmega()

        var_dailyIntake = obj1.getdailyIntake()
        var_kordaP2evas = obj1.getkordaP2evas()

        var_kellaaeg1= obj1.getkellaaeg1()
        var_kellaaeg2= obj1.getkellaaeg2()
        var_kellaaeg3= obj1.getkellaaeg3()
        var_kellaaeg4= obj1.getkellaaeg4()
        var_kellaaeg5= obj1.getkellaaeg5()
        var_kellaajad=obj1.getkellaajad()

        flash('Settings saved!')

    else:
        flash('Error: All the form fields are required. ')

  return render_template('loputooTemplate.v0.6.html', form=form,
                    var_dailyIntake=obj1.getdailyIntake(),
                    var_kordaP2evas=obj1.getkordaP2evas(),
                    var_kellaaeg1=obj1.getkellaaeg1(),
                    var_kellaaeg2=obj1.getkellaaeg2(),
                    var_kellaaeg3=obj1.getkellaaeg3(),
                    var_kellaaeg4=obj1.getkellaaeg4(),
                    var_kellaaeg5=obj1.getkellaaeg5(),
                    var_kellaajad=obj1.getkellaajad()
                    )

if __name__ == '__main__':
  app.run(debug=True, port=8080, host='0.0.0.0', use_reloader=False)

time.sleep(1)
GPIO.output(7, False)
GPIO.cleanup()
```

LISA L arendusplaadi ja *Raspberry Pi* kommunikatsiooni kood. Veebilehe vormingu osa

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html>
   <head>
      <title>Reusable Form Demo - https://pythonspot.com/en/flask-web-forms </title>
      <link rel="stylesheet" media="screen" href ="static/bootstrap.min.css">
      <link rel="stylesheet" href="static/bootstrap-theme.min.css">
      <meta name="viewport" content = "width=device-width, initial-scale=1.0">

   </head>
   <body>

<div class="container">
  <h2>Current time</h2>
  <form  action="" method="post" role="form">
   {{ form.csrf }}
   <div class="form-group">

     hr. |min
     <br>
     <input type="text" class="form-control" id="hour" name="hour" placeholder="00"
size="1">
     <input type="text" class="form-control" id="minutes" name="minutes"
placeholder="00" size="1">

     <br><br>
     <label for="name">Daily food intake in grams: </label>
     <br>
     <input type="text" class="form-control" id="dailyIntake" name="dailyIntake"
placeholder="" size="3" value={{ var_dailyIntake }}>
     <br>
     <br>
     <label for="name">Times per day: </label>
     <br>
     <input type="text" class="form-control" id="kordaP2evas" name="kordaP2evas"
placeholder="3" size="2" value={{ var_kordaP2evas }}>
     <br>
     <br>
     <label for="name">At following times (24h format)
     <br>(max 5 times):</label>
     <br>
     <input type="text" class="form-control" id="kellaaegadel" name="kellaaegadel"
placeholder="06,12,18,21,03" size="14" value={{ var_kellaajad }} >
     <br>
```

LISA L arendusplaadi ja *Raspberry Pi* kommunikatsiooni kood. Veebilehe vormingu osa (jätk)

```
    </div>
    <br>
    <button type="submit" class="btn btn-success">Save</button>
  </form>

  <br>
      {% with messages = get_flashed_messages(with_categories=true) %}
         {% if messages %}

      {% for message in messages %}
         {% if "Error" not in message[1]: %}
            <div class="alert alert-info">
            <strong>Success! </strong> {{ message[1] }}

            </div>
         {% endif %}

         {% if "Error" in message[1]: %}
            <div class="alert alert-warning">
            {{ message[1] }}
            </div>
         {% endif %}
      {% endfor %}
         {% endif %}
      {% endwith %}

</div>
<br>
</div>
</div>
</body>
</html>
```

# LISA M

LISA M *16 bit ADC* põhiprogrammi kood

```
#define F_CPU 8000000UL
#include <avr/io.h>
#include <util/delay.h>
#include <stdlib.h>
#include <avr/sleep.h>
#include <avr/power.h>
#include <avr/interrupt.h>
#include " UART_lib_v0.5_2Mhz.h" [LISA K]
#include "I2C_lib.v.0.2_gague.h"

enum{averageArrSize = 10};
double          averageArr[averageArrSize];
double          samples         = 0;
uint16_t        sample          = 0;
uint32_t        adcSum          = 0;
double          averageSum      = 0;
double          average         = 0;
double          averagefloored  = 0;
double          previousAverage = 0;
char            outstr[15];
char            buffer[23];
int             radix    = 10;

double averageOf_X_Samples(double samples, int debug){

        adcSum                  =  0;
        for (int i=0; i < samples; i++){
                sample = _I2C_ReadReg(0b11010001);

                if (debug == 1){
                        USART_SendString("sample:                       ");
                        ltoa(sample,buffer,radix);
                        USART_SendString(buffer);
                        USART_sendNewline();
                }
                adcSum = adcSum + sample;
        }

        if (debug == 1){
                USART_SendString("adcSum:                       ");
                ltoa(adcSum,buffer,radix);
                USART_SendString(buffer);
                USART_sendNewline();
        }

        double average = adcSum / samples;
```

LISA M *16 bit ADC* põhiprogrammi kood (jätk)

```
	return average;
}

double takeAnAverageFromAverageArray(int debug){

	averageSum    =  0;
	for (int i=0; i < averageArrSize; i++){
		averageSum = averageSum + averageArr[i];

			if (debug == 1){
				dtostrf(averageArr[i], 11, 3, outstr);
				USART_SendString("averageArr[i]:  ");
				USART_SendString(outstr);
				USART_sendNewline();
			}
	}
	average = averageSum / (double) averageArrSize;
	return average;
}

int main(void){

	clock_prescale_set(clock_div_2);

	DDRC = (1<<DDC6)|(1<<DDC7);
	PORTC = (1<<DDC6);
	_delay_ms(200);

	USART_init();
	TWIInit(); /*


	_I2C_WriteReg(0b11010000, 0b000011000);
	int debug = 0;
	while(1){

		for (int i=0; i < averageArrSize; i++){
	average = averageOf_X_Samples(50, 0);
			averageArr[i] = average;

			if (debug == 1){
				dtostrf(average, 11, 3, outstr);
				USART_SendString("Average:  ");
				USART_SendString(outstr);
				USART_sendNewline();
			}
		}
```

LISA M *16 bit ADC* põhiprogrammi kood (jätk)

```
                average = takeAnAverageFromAverageArray(0);
                dtostrf(average, 11, 3, outstr);
                USART_SendString("averageOfaverageArr:"); USART_sendChar('
        ');                     USART_SendString(outstr);
                USART_sendNewline();


        }
}
```

LISA M *16 bit ADC* jadaliidese alamprogrammi kood

```
#include <util/delay.h>
#include <stdlib.h>

void _USARTSend_Status(uint8_t, char[]);
void USART_send(char);
void USART_send_byte(uint8_t);
void USART_send_2x_byte(uint16_t);

void TWIInit(){

	TWSR = 0x00;
	TWBR = 0x0C;

	TWCR = (1<<TWEN);
}

void _infiniteLoop(){

	DDRC = (1<<DDC7);

	while(1)
	{
		PINC = (1<<PINC7);
		_delay_ms(1000);
	}
}

void I2CWaitFor_TWINT_FlagToBeSet(){

	while((TWCR & (1<<TWINT)) == 0);
}

void I2CCheckStatusRegister(uint8_t Status){

	if ((TWSR & 0xF8) != Status){
		if (Status == 0x08){
			USART_SendString("START condition was not sent!");
		}else if ((TWSR & 0xF8) == 0x20){
			USART_SendString("ERROR: SLA+W has been transmitted; NOT
ACK has been received, Status code 0x20");
		}
		USART_sendNewline();

		while(1){ _delay_ms(100); }
	}
}
```

LISA M *16 bit ADC* jadaliidese alamprogrammi kood (jätk)

```
void I2C_Start(){

        TWCR = (1<<TWINT)|(1<<TWSTA)|(1<<TWEN);
        while ((TWCR & (1<<TWINT)) == 0);

}

void I2C_ClearFlagAndEnable(){

        TWCR = (1<<TWINT)|(1<<TWEN);
}

uint8_t _I2cGetTWSR(){

        uint8_t status;
        status = TWSR & 0xF8;
        return status;
}

void _I2C_TestConnectionToDevice(uint8_t Slaveaddress){

        I2C_Start();
        I2CWaitFor_TWINT_FlagToBeSet();
        I2CCheckStatusRegister(0x08);
        USART_SendString("START condition transmitted");
        USART_sendNewline();

        TWDR = Slaveaddress;
        I2C_ClearFlagAndEnable();
        I2CWaitFor_TWINT_FlagToBeSet();
        I2CCheckStatusRegister(0x18);
        USART_SendString("SLA+W sent, ACK received");
        USART_sendNewline();

        TWCR  =(1<<TWEN)|(1<<TWSTO)|(1<<TWINT);
        USART_SendString("STOP condition transmitted");
        USART_sendNewline();
}

void _I2C_WriteReg(uint8_t Slaveaddress, uint8_t ConfigurationByte){

        USART_SendString("_I2C_WriteReg START"); USART_sendNewline();

        I2C_Start();
        I2CWaitFor_TWINT_FlagToBeSet();
        I2CCheckStatusRegister(0x08);
```

LISA M *16 bit ADC* jadaliidese alamprogrammi kood (jätk)

```
        TWDR = Slaveaddress;
        I2C_ClearFlagAndEnable();
        I2CWaitFor_TWINT_FlagToBeSet();
        I2CCheckStatusRegister(0x18);

        TWDR = ConfigurationByte;
        I2C_ClearFlagAndEnable();
        I2CWaitFor_TWINT_FlagToBeSet();
        I2CCheckStatusRegister(0x28);

        TWCR  =(1<<TWEN)|(1<<TWSTO)|(1<<TWINT);

        USART_SendString("_I2C_WriteReg STOP");
        USART_sendNewline();
}

uint16_t _I2C_ReadReg(uint8_t Slaveaddress){

        uint8_t Databyte1 = 0;
        uint8_t Databyte2 = 0;
        uint16_t c = 0;

        I2C_Start();
        I2CWaitFor_TWINT_FlagToBeSet();

        I2CCheckStatusRegister(0x08);
        TWDR = Slaveaddress;
        I2C_ClearFlagAndEnable();
        I2CWaitFor_TWINT_FlagToBeSet();
        I2CCheckStatusRegister(0x40);

        TWCR = (1<<TWINT)|(1<<TWEN)|(1<<TWEA);
        while ((TWCR & (1<<TWINT)) == 0);
        Databyte1 = TWDR;

        TWCR = (1<<TWINT)|(1<<TWEN)|(1<<TWEA);
        while ((TWCR & (1<<TWINT)) == 0);
        Databyte2 = TWDR;

        TWCR = (1<<TWINT)|(1<<TWEN)|(1<<TWEA);
        while ((TWCR & (1<<TWINT)) == 0);

        TWCR = (1<<TWINT)|(1<<TWEN)|(0<<TWEA);
        while ((TWCR & (1<<TWINT)) == 0);

        c = (Databyte1 << 8);
        c |= Databyte2;
```

LISA M *16 bit ADC* jadaliidese alamprogrammi kood (jätk)

```
        TWCR  =(1<<TWEN)|(1<<TWSTO)|(1<<TWINT);

        return c;
}
```

**Lisa 5. Lihtlitsents lõputöö salvestamiseks ja üldsusele kättesaadavaks tegemiseks ning juhendaja(te) kinnitus lõputöö kaitsmisele lubamise kohta**

Mina, Kaljo Valk,
(*autori nimi*)
sünniaeg 1982,

1. annan Eesti Maaülikoolile tasuta loa (lihtlitsentsi) enda loodud lõputöö

Automaatne kuiva kassitoidu jaotur,
(*lõputöö pealkiri*)

mille juhendaja(d) on Erkki Jõgi,
(*juhendaja(te) nimi*)

1.1. salvestamiseks säilitamise eesmärgil,
1.2. digiarhiivi DSpace lisamiseks ja
1.3. veebikeskkonnas üldsusele kättesaadavaks tegemiseks

kuni autoriõiguse kehtivuse tähtaja lõppemiseni;

2. olen teadlik, et punktis 1 nimetatud õigused jäävad alles ka autorile;
3. kinnitan, et lihtlitsentsi andmisega ei rikuta teiste isikute intellektuaalomandi ega isikuandmete kaitse seadusest tulenevaid õigusi.

Lõputöö autor _____
(*allkiri*)

Tartu, 24.05.2017
(*kuupäev*)

---

**Juhendaja(te) kinnitus lõputöö kaitsmisele lubamise kohta**

Luban lõputöö kaitsmisele.

Erkki Jõgi — 24.05.2017
(*juhendaja nimi ja allkiri*) — (*kuupäev*)

_____ — _____
(*juhendaja nimi ja allkiri*) — (*kuupäev)*